2 線性方程式及特徵系統

2-1 2-2 2-3 2-4 2-5 2-6 2-7 2-8 2-9 2-10 2-11 2-12 2-13 2-14 2-15 2-16 2-17

- ❖ MATLAB 是研究線性方程式系統的理想作業環境。
- ❖ 2000 年MATLAB 開始用線性代數子程序的LAPACK 函式庫。

2.1 導論

❖ 以一簡單電子電路中迴路電流的計算為例。

❖ 迴路電流如圖2.1 所示

$$V_{bc} = R_2(I_1 - I_2)$$

$$V_{ab} + V_{bc} + V_{cd} = V$$

$$R_1 I_1 + R_2(I_1 - I_2) + R_4 I_1 = V$$

▶數值方法－使用MATLAB程式語言(第三版)

2-1 2-2 2-3 2-4 2-5 2-6 2-7 2-8 2-9 2-10 2-11 2-12 2-13 2-14 2-15 2-16 2-17

❖ 得出以下四個方程式

$$\begin{aligned}(R_1 + R_2 + R_4) I_1 - R_2 I_2 &= V \\ (R_1 + 2R_2 + R_4) I_2 - R_2 I_1 - R_2 I_3 &= 0 \\ (R_1 + 2R_2 + R_4) I_3 - R_2 I_2 - R_2 I_4 &= 0 \\ (R_1 + R_2 + R_3 + R_4) I_4 - R_2 I_3 &= 0\end{aligned} \qquad (2.1)$$

❖ 令 $R_1 = R_4 = 1\Omega, R_2 = 2\Omega, R_3 = 4\Omega,$ 及 $V = 5\,\mathrm{V}$，(2.1) 變成

$$4I_1 - 2I_2 = 5$$
$$-2I_1 + 6I_2 - 2I_3 = 0$$
$$-2I_2 + 6I_3 - 2I_4 = 0$$
$$-2I_3 + 8I_4 = 0$$

❖ 以矩陣的符號表示

$$\begin{bmatrix} 4 & -2 & 0 & 0 \\ -2 & 6 & -2 & 0 \\ 0 & -2 & 6 & -2 \\ 0 & 0 & -2 & 8 \end{bmatrix} \begin{bmatrix} I_1 \\ I_2 \\ I_3 \\ I_4 \end{bmatrix} = \begin{bmatrix} 5 \\ 0 \\ 0 \\ 0 \end{bmatrix} \qquad (2.2)$$

這方程式的型為**Ax=b**

▶數值方法－使用MATLAB程式語言(第三版)

2-1 2-2 2-3 2-4 2-5 2-6 2-7 2-8 2-9 2-10 2-11 2-12 2-13 2-14 2-15 2-16 2-17

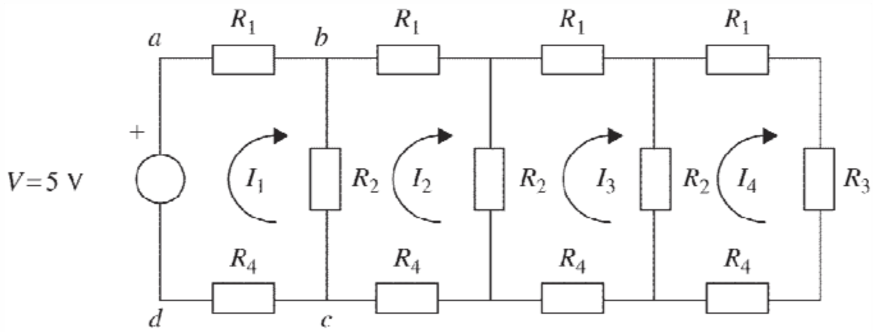

圖2.1　電子電路。

```
>> A = [4 -2 0 0;-2 6 -2 0;0 -2 6 -2;0 0 -2 8];
>> b = [5 0 0 0].';
>> A\b

ans =
    1.5426
    0.5851
    0.2128
    0.0532
```


❖ 我們將圖2.1 的網路5V 直流換成5V 交流，理想電阻$R_1, \cdots R_4$ 以阻抗$Z_1 \cdots Z_4$ 取代，所以(2.1) 成為

$$
\begin{aligned}
(Z_1+Z_2+Z_4)I_1 - Z_2I_2 &= V \\
(Z_1+2Z_2+Z_4)I_2 - Z_2I_1 - Z_2I_3 &= 0 \\
(Z_1+2Z_2+Z_4)I3 - Z_2I_2 - Z_2I_4 &= 0 \\
(Z_1+Z_2+Z_3+Z_4)I_4 - Z_2I_3 &= 0
\end{aligned}
\qquad (2.3)
$$

❖ 假設Z_1=Z_4=(1+0.5j)，Z_2=(2+0.5j) 與Z_3=(4+1j)，此處j= $\sqrt{-1}$

❖ 所以(2.3) 變成

$$\begin{aligned}(4+1.5j)I_1-(2+0.5j)I_2&=5\\-(2+0.5j)I_1+(6+2.0j)I_2-(2+0.5j)I_3&=0\\-(2+0.5j)I_2+(6+2.0j)I_3-(2+0.5j)I_4&=0\\-(2+0.5j)I_3+(8+2.5j)I_4&=0\end{aligned}$$

$$\begin{bmatrix}(4+1.5j)&-(2+0.5j)&0&0\\-(2+0.5j)&(6+2.0j)&-(2+0.5j)&0\\0&-(2+0.5j)&(6+2.0j)&-(2+0.5j)\\0&0&-(2+0.5j)&(8+2.5j)\end{bmatrix}\begin{bmatrix}I_1\\I_2\\I_3\\I_4\end{bmatrix}=\begin{bmatrix}5\\0\\0\\0\end{bmatrix}\tag{2.4}$$

```
>> p = 4+1.5i; q = -2-0.5i;
>> r = 6+2i; s = 8+2.5i;
>> A = [p q 0 0;q r q 0;0 q r q;0 0 q s];
>> b = [5 0 0 0].';
>> A\b

ans =
   1.3008 - 0.5560i
   0.4560 - 0.2504i
   0.1530 - 0.1026i
   0.0361 - 0.0274i
```


2.2 線性方程式系統

- 一個線性方程式可以寫成如下的矩陣形式

$$\mathbf{Ax} = \mathbf{b} \tag{2.5}$$

- 若**b**=**0**，則(2.5) 式稱為齊次式，若**b** ≠ 0 則稱為非齊次式。
- 由圖2.2可說明解的性質：唯一解、無限多解或無解。
- (2.5) 式的非齊次方程式

$$\mathbf{A}^{-1}\mathbf{Ax} = \mathbf{A}^{-1}\mathbf{b} \tag{2.6}$$

$$\mathbf{A}^{-1}\mathbf{A} = \mathbf{A}\mathbf{A}^{-1} = \mathbf{I} \tag{2.7}$$

$$\mathbf{x} = \mathbf{A}^{-1}\mathbf{b} \tag{2.8}$$

$$\mathbf{A}^{-1} = \mathrm{adj}(\mathbf{A})/|\mathbf{A}| \tag{2.9}$$

▶ 數值方法－使用MATLAB程式語言(第三版)

伴隨矩陣

2-1
2-2
2-3
2-4
2-5
2-6
2-7
2-8
2-9
2-10
2-11
2-12
2-13
2-14
2-15
2-16
2-17

2x2矩陣

一個2×2矩陣 $\mathbf{A} = \begin{pmatrix} a & b \\ c & d \end{pmatrix}$的伴隨矩陣是

$$\mathrm{adj}(\mathbf{A}) = \begin{pmatrix} d & -b \\ -c & a \end{pmatrix}.$$

3x3矩陣

對於3×3的矩陣，情況稍微複雜一點：

$$\mathbf{A} = \begin{pmatrix} A_{11} & A_{12} & A_{13} \\ A_{21} & A_{22} & A_{23} \\ A_{31} & A_{32} & A_{33} \end{pmatrix}.$$

其伴隨矩陣是：

$$\mathrm{adj}(\mathbf{A}) = \begin{pmatrix} +\begin{vmatrix} A_{22} & A_{23} \\ A_{32} & A_{33} \end{vmatrix} & -\begin{vmatrix} A_{12} & A_{13} \\ A_{32} & A_{33} \end{vmatrix} & +\begin{vmatrix} A_{12} & A_{13} \\ A_{22} & A_{23} \end{vmatrix} \\ -\begin{vmatrix} A_{21} & A_{23} \\ A_{31} & A_{33} \end{vmatrix} & +\begin{vmatrix} A_{11} & A_{13} \\ A_{31} & A_{33} \end{vmatrix} & -\begin{vmatrix} A_{11} & A_{13} \\ A_{21} & A_{23} \end{vmatrix} \\ +\begin{vmatrix} A_{21} & A_{22} \\ A_{31} & A_{32} \end{vmatrix} & -\begin{vmatrix} A_{11} & A_{12} \\ A_{31} & A_{32} \end{vmatrix} & +\begin{vmatrix} A_{11} & A_{12} \\ A_{21} & A_{22} \end{vmatrix} \end{pmatrix}$$

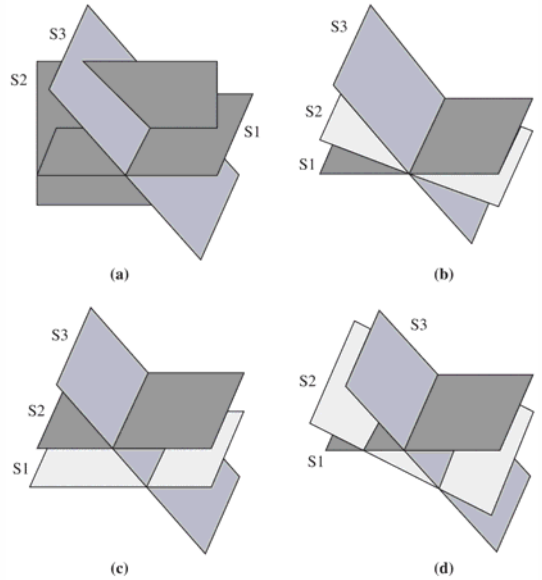

圖2.2　三變數之三個平面方程式的交會情形。

- 此處|**A**|代表A 的行列式，adj(**A**) 是**A** 的伴隨矩陣(adjoint)。
- 在|**A**| = 0 的條件下，**A** 沒有反矩陣。
- 這時矩陣**A** 稱為奇異矩陣(singular) 且**x** 的唯一解不存在。
- 矩陣的位階(rank)：對一方陣而言，位階是矩陣中獨立的列數或行數。

2-1 2-2 2-3 2-4 2-5 2-6 2-7 2-8 2-9 2-10 2-11 2-12 2-13 2-14 2-15 2-16 2-17

$$\begin{bmatrix} 1 & 2 & 3 \\ -2 & 1 & 4 \\ -1 & 3 & 4 \end{bmatrix} \text{ 或 } \begin{bmatrix} [1 & 2 & 3] \\ [-2 & 1 & 4] \\ [-1 & 3 & 7] \end{bmatrix} \text{ 或 } \left[\begin{bmatrix} 1 \\ 2 \\ 3 \end{bmatrix} \begin{bmatrix} -2 \\ 1 \\ 4 \end{bmatrix} \begin{bmatrix} -1 \\ 3 \\ 7 \end{bmatrix} \right]$$

$$\begin{bmatrix} 1 & 2 & 3 \\ 2 & 4 & 6 \\ 3 & 6 & 9 \end{bmatrix}$$

2-1 2-2 2-3 2-4 2-5 2-6 2-7 2-8 2-9 2-10 2-11 2-12 2-13 2-14 2-15 2-16 2-17

❖ 矩陣通常不為方陣，$m \times n$ 階矩陣**A** 的位階記為rank(A)，若rank(**A**) = min(m,n) 則稱為滿位階(full rank)，若rank(**A**) < min(m,n) 則矩陣**A** 稱為位階不足(rank deficient)。

2-1
2-2
2-3
2-4
2-5
2-6
2-7
2-8
2-9
2-10
2-11
2-12
2-13
2-14
2-15
2-16
2-17

```
>> D = [1 2 3;3 4 7;4 -3 1;-2 5 3;1 -7 6]

D =
     1     2     3
     3     4     7
     4    -3     1
    -2     5     3
     1    -7     6

>> rank(D)

ans =
     3
```


2-1
2-2
2-3
2-4
2-5
2-6
2-7
2-8
2-9
2-10
2-11
2-12
2-13
2-14
2-15
2-16
2-17

❖ 線性代數中的一個有效運算是將矩陣轉化成矩陣的梯式簡化列(reduced row echelon form;RREF)。

```
>> rref(D)

ans =
     1     0     0
     0     1     0
     0     0     1
     0     0     0
     0     0     0
```

- 矩陣的RREF 的特性是非零的列數等於矩陣的位階。
- 令**A** 是$n \times n$ 階的矩陣，若**Ax**=**b** 是一致的且有唯一解，則：
  - **Ax**=**0** 有唯一的無關緊要解(trivial solution) **x**=**0**
  - **A** 是非奇異性(non−singular) 且det(**A**) ≠ 0
  - **A** 的RREF 是單位矩陣
  - **A** 具有n 個獨立列及行
  - **A** 為滿位階(full rank)，即rank(**A**) = n

- 若**Ax** = b 不是一致性的就是非一致性的但不僅只有一解，則：
  - **Ax** = 0 有不只一解
  - **A** 是奇異的且det(**A**) = 0
  - **A** 的RREF 最少有一列為零
  - **A** 有線性相關的列及行
  - **A** 是位階不足(rank deficient)，即rank(**A**) < n

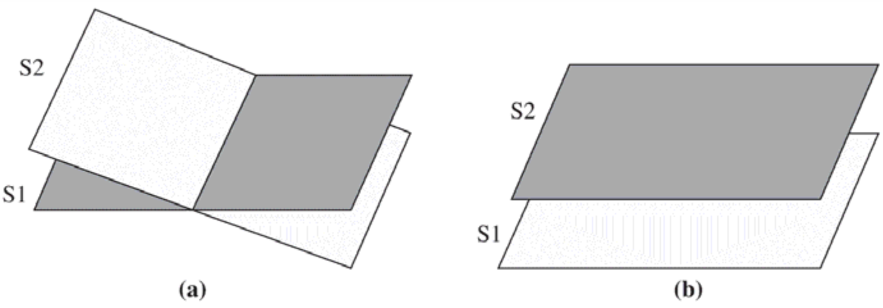

圖2.3　表示低於所求型方程組的平面 (a) 兩平面相交於一線 (b) 兩平面無相交。

▶ 數值方法－使用MATLAB程式語言(第三版)

❖ 考慮以下方程式系統：

$$\begin{bmatrix} 1 & 2 & 3 & 4 \\ -4 & 2 & -3 & 7 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ x_3 \\ x_4 \end{bmatrix} = \begin{bmatrix} 1 \\ 3 \end{bmatrix}$$

❖ 這系統屬於低於所求型，重新排列如下：

$$\begin{bmatrix} 1 & 2 \\ -4 & 2 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} + \begin{bmatrix} 3 & 4 \\ -3 & 7 \end{bmatrix} \begin{bmatrix} x_3 \\ x_4 \end{bmatrix} = \begin{bmatrix} 1 \\ 3 \end{bmatrix}$$

或

$$\begin{bmatrix} 1 & 2 \\ -4 & 2 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} = \begin{bmatrix} 1 \\ 3 \end{bmatrix} - \begin{bmatrix} 3 & 4 \\ -3 & 7 \end{bmatrix} \begin{bmatrix} x_3 \\ x_4 \end{bmatrix}$$

2-1
2-2
2-3
2-4
2-5
2-6
2-7
2-8
2-9
2-10
2-11
2-12
2-13
2-14
2-15
2-16
2-17

❖若系統的方程式較未知數多，則此系統稱為高於所求型或供過於求型(over–determined)。圖2.4 表示3 度空間內的4 個平面，代表有3 個變數的4 個平面方程式。

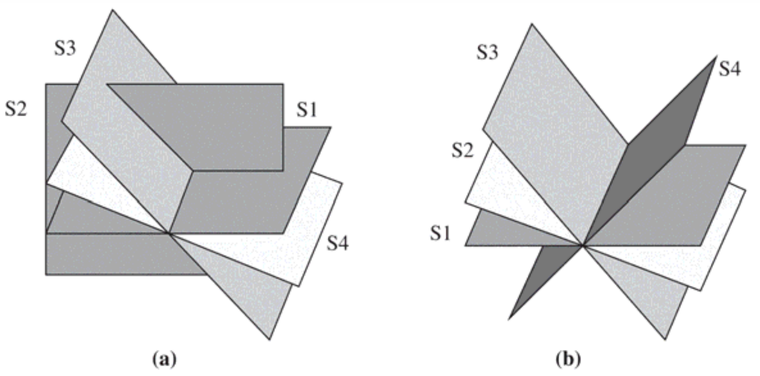

圖2.4　表示高於所求型方程組的平面 (a) 四個平面相交於一個點 (b) 四個平面相交於一線

2-1 2-2 2-3 2-4 2-5 2-6 2-7 2-8 2-9 2-10 2-11 2-12 2-13 2-14 2-15 2-16 2-17

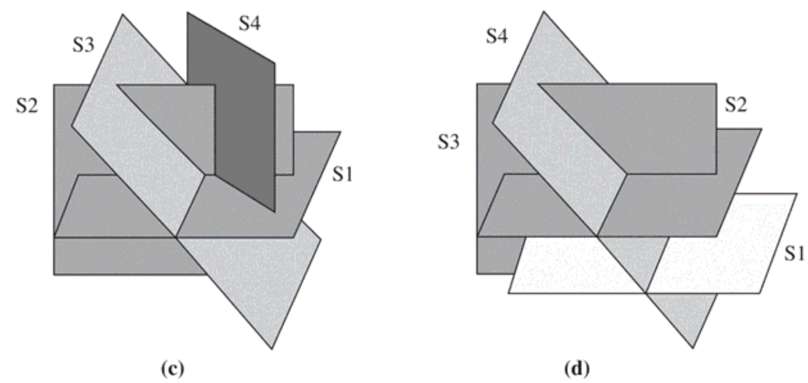

圖2.4　(c) 四個平面未相交於一點，只可看到(S1, S2, S3) 和(S1, S2, S4) 的相交點 (d) 四個平面表示不相容方程式。

2.3　解Ax=b 之MATLAB 運算元＼及／

2-1 2-2 2-3 2-4 2-5 2-6 2-7 2-8 2-9 2-10 2-11 2-12 2-13 2-14 2-15 2-16 2-17

❖ 運算元／和＼完成矩陣除法且有同樣效果。

❖ 求解**Ax**= **b**。

❖ 可以寫x=A\b 或x'=b'/A'。

❖ 解**Ax** = **b** 時，運算元／或＼依矩陣**A** 之特性選擇適當的演算法。這些情形摘要如下：

- 若(if)

  A是三角矩陣則單獨以後向(back) 或前向(forward) 代入法求解，2.6 節做說明。

- •否則(elseif)

  A 若是正定(positive definite)，方陣對稱(square symmetric) 或是Hermitian 矩陣，2.8 節中將說明使用柯列斯基(Cholesky) 分解。若**A** 是稀疏矩陣(sparse)，對稱性最低階數預排(preordering) 必須先完成再配合柯列斯基分解(2.14 節將做說明)。

- •否則(elseif)

  A若是方陣， 使用一般的LU 分解(2.7 節說明)， **A** 若是稀疏矩陣，則須先完成非對稱性最低階數預排(non–symmetric minimum degreepreordering)。

2-1 2-2 2-3 2-4 2-5 2-6 2-7 2-8 2-9 2-10 2-11 2-12 2-13 2-14 2-15 2-16 2-17

▶ 數值方法－使用MATLAB程式語言(第三版)

- •否則(elseif)

  A若是滿非方陣型(full non–square) 矩陣，應用QR 分解(2.9 節說明)。

- •否則(elseif)

  A 若是稀疏的非方陣型，則以擴大矩陣(augmented) 及最低階數預排配合稀疏高斯消去法(sparse Gaussian elimination，2,14 節做說明)。

❖MATLAB 提供一內建函數inv(A) 來求取矩陣的反矩陣。

2-1 2-2 2-3 2-4 2-5 2-6 2-7 2-8 2-9 2-10 2-11 2-12 2-13 2-14 2-15 2-16 2-17

❖檢驗運算元\對滿矩陣的運算時間並與該矩陣經適當補零為三角矩陣的結果比較

```
% e3s201.m
disp('    n     full-time  full-time/n^3    tri-time   tri-time/n^2');
A = [ ]; b = [ ];
for n = 2000:500:6000
    A = 100*rand(n); b = [1:n].';
    tic, x = A\b; t1 = toc;
    t1n = 5e9*t1/n^3;
    for i = 1:n
        for j = i+1:n
            A(i,j) = 0;
        end
    end
    tic, x = A\b; t2 = toc;
    t2n = 1e9*t2/n^2;
    fprintf('%6.0f %9.4f %12.4f %12.4f %11.4f\n',n,t1,t1n,t2,t2n)
end
```


n	full-time	full-time/n^3	tri-time	tri-time/n^2
2000	1.7552	1.0970	0.0101	2.5203
2500	3.3604	1.0753	0.0151	2.4151
3000	5.4936	1.0173	0.0209	2.3275
3500	8.5735	0.9998	0.0282	2.3001
4000	12.6882	0.9913	0.0358	2.2393
4500	17.5680	0.9639	0.0453	2.2392
5000	24.8408	0.9936	0.0718	2.8703

正定矩陣

設 A 爲 $n \times n$ 階實矩陣，若任意 n 維非零向量 $\mathbf{x}$ 滿足

$$\mathbf{x}^T A\mathbf{x} > 0，$$

我們稱 A 爲正定 (positive definite)；將上述條件放鬆爲

$$\mathbf{x}^T A\mathbf{x} \geq 0，$$

則 A 稱作半正定 (positive semidefinite)。改變正定和半正定的不等式方向就有 A 是負定或半負定的概念，也可以說 $-A$ 是正定或半正定。如果 $\mathbf{x}^T A\mathbf{x}$ 可能是正值也可能是負值，則稱 A 是未定的 (indefinite)。

❖ 檢驗運算元＼對於正定對稱系統(positive definite symmetric systems) 的影響。x*Ax > 0

❖ 程式如下：

2-1
2-2
2-3
2-4
2-5
2-6
2-7
2-8
2-9
2-10
2-11
2-12
2-13
2-14
2-15
2-16
2-17

```
% e3s202.m
disp(' n       time-pos    time-pos/n^3  time-npos    time-b/n^3');
for n = 100:100:1000
    A = [ ]; M = 100*randn(n,n);
    A = M*M'; b = [1:n].';
    tic, x = A\b; t1 = toc*1000;
    t1d = t1/n^3;
    A = A+rand(size(A));
    tic, x = A\b; t2 = toc*1000;
    t2d = t2/n^3;
    fprintf('%4.0f %10.4f %14.4e %11.4f %13.4e\n',n,t1,t1d,t2,t2d)
end
```

n	time-pos	time-pos/n^3	time-npos	time-b/n^3
100	0.9881	9.8811e-007	1.2085	1.2085e-006
200	3.5946	4.4932e-007	3.0903	3.8629e-007
300	5.0646	1.8758e-007	9.7878	3.6251e-007
400	10.3890	1.6233e-007	20.4892	3.2014e-007
500	18.0235	1.4419e-007	36.5653	2.9252e-007
600	18.1892	8.4209e-008	37.7766	1.7489e-007
700	26.5483	7.7400e-008	58.3854	1.7022e-007
800	39.6402	7.7422e-008	79.4285	1.5513e-007
900	58.5519	8.0318e-008	110.5409	1.5163e-007
1000	67.9078	6.7908e-008	130.2029	1.3020e-007

Hilbert 矩陣

❖ 在線性代數中，希爾伯特矩陣是一種係數都是單位分數的方塊矩陣。具體來說一個希爾伯特矩陣H的第i橫行第j縱列的係數是：

$$H_{ij} = \frac{1}{i+j-1}.$$

舉例來說，5×5的希爾伯特矩陣就是：

$$H_5 = \begin{bmatrix} 1 & \frac{1}{2} & \frac{1}{3} & \frac{1}{4} & \frac{1}{5} \\ \frac{1}{2} & \frac{1}{3} & \frac{1}{4} & \frac{1}{5} & \frac{1}{6} \\ \frac{1}{3} & \frac{1}{4} & \frac{1}{5} & \frac{1}{6} & \frac{1}{7} \\ \frac{1}{4} & \frac{1}{5} & \frac{1}{6} & \frac{1}{7} & \frac{1}{8} \\ \frac{1}{5} & \frac{1}{6} & \frac{1}{7} & \frac{1}{8} & \frac{1}{9} \end{bmatrix}.$$

- ❖ 的檢驗運算元＼對條件數惡劣的Hilbert 矩陣仍能成功的運用。
- ❖ 程式如下：

```
% e3s203.m
disp(' n  time-slash acc-slash   time-inv   acc-inv   condition');
for n = 4:2:20
    A = hilb(n); b = [1:n].';
    tic, x = A\b; t1 = toc; t1 = t1*10000;
    nm1 = norm(b-A*x);
    tic, x = inv(A)*b; t2 = toc; t2 = t2*10000;
    nm2 = norm(b-A*x);
    c = cond(A);
    fprintf('%2.0f %10.4f %10.2e %8.4f %11.2e %11.2e \n',n,t1,nm1,t2,nm2,c)
end
```


n	time-slash	acc-slash	time-inv	acc-inv	condition
4	1.6427	1.39e-013	0.8549	9.85e-014	1.55e+004
6	0.9415	5.22e-012	0.7710	2.02e-009	1.50e+007
8	1.1454	5.35e-010	0.8465	3.19e-006	1.53e+010
10	1.2627	3.53e-008	1.5477	2.47e-004	1.60e+013
12	1.9332	1.40e-006	1.5589	9.39e-001	1.74e+016
14	2.1958	3.36e-005	1.5924	3.39e+002	5.13e+017
16	2.3187	5.76e-006	1.6650	1.02e+002	4.52e+017
18	2.4836	5.25e-005	2.0589	2.31e+002	1.57e+018
20	2.4417	1.11e-005	2.0869	3.72e+002	2.57e+018

2.4　解的精確度及病態條件

範例2.1

❖ 考慮下列MATLAB 敘述

```
>> A =  3.021 2.714 6.913;1.031 -4.273 1.121;5.084 -5.832 9.155

A =
    3.0210    2.7140    6.9130
    1.0310   -4.2730    1.1210
    5.0840   -5.8320    9.1550

>> b = [12.648 -2.121 8.407].'

b =
   12.6480
   -2.1210
    8.4070
```


範例2.1

```
>> A\b

ans =
    1.0000
    1.0000
    1.0000
```

❖ 這結果是正確的，代回原問題可以很容易檢驗。

範例2.2

❖ 考慮例2.1 中將A (2,2) 由−4.2730 改為−4.2750

```
>> A(2,2) = -4.2750

A =
    3.0210    2.7140    6.9130
    1.0310   -4.2750    1.1210
    5.0840   -5.8320    9.1550

>> A\b

ans =
   -1.7403
    0.6851
    2.3212
```

❖ 雖然在例2.1 中A(2,2) 係數的變化率小於0.1%，但結果與例2.1 卻差異甚大。

▶ 數值方法－使用MATLAB程式語言(第三版)

2-1 2-2 2-3 2-4 2-5 2-6 2-7 2-8 2-9 2-10 2-11 2-12 2-13 2-14 2-15 2-16 2-17

解的精確度及病態條件

❖ 假如係數矩陣A的細微變化造成差異甚大的解答，此方程式系統被稱為是病態。

❖ MATLAB 提供2 個內建函數cond 與rcond 來估測矩陣狀態。

❖ cond函數是一種奇異值分解(singular value decomposition)

❖ 對完美狀態矩陣cond是1，但若是病態矩陣則為一個大數值

❖ rcod函數較不可靠但較快，此函數給出0與1間的數值，數值越小，矩陣愈是病態，而rcond的倒數通常大小與cond差不多

範例2.3

❖ 完美狀態矩陣的說明：

```
>> A = diag([20 20 20])
A =
    20     0     0
     0    20     0
     0     0    20

>> [det(A) rcond(A) cond(A)]

ans =
        8000             1            1
```


範例2.4

❖ 病態矩陣的說明

```
>> A = [1 2 3;4 5 6;7 8 9.000001];
>> format short e
>> [det(A) rcond(A) 1/rcond(A) cond(A)]

ans =
  -3.0000e-006  6.9444e-009  1.4400e+008  1.0109e+008
```

rcond 的倒數與cond 之數值很接近。

❖ 以MATLAB函數cond及rcond來研究Hilbert矩陣的條件數

```
% e3s204.m Hilbert matrix test.
disp('     n           cond            rcond        log10(cond)')
for n = 4:2:20
    A = hilb(n);
    fprintf('%5.0f %16.4e',n,cond(A));
    fprintf('%16.4e %10.2f\n',rcond(A),log10(cond(A)));
end
```


n	cond	rcond	log10(cond)
4	1.5514e+004	3.5242e-005	4.19
6	1.4951e+007	3.4399e-008	7.17
8	1.5258e+010	2.9522e-011	10.18
10	1.6025e+013	2.8286e-014	13.20
12	1.7352e+016	2.6328e-017	16.24
14	5.1317e+017	1.7082e-019	17.71
16	4.5175e+017	4.6391e-019	17.65
18	1.5745e+018	5.8371e-020	18.20
20	2.5710e+018	1.9953e-019	18.41

❖ 函數呼叫的範例如下：

```
gallery('hanowa',6,4)
gallery('cauchy',6)
gallery('forsythe',6,8)
```


2.5 基本列運算

❖ 系統具有如下形式

$$
\begin{aligned}
a_{11}x_1 + a_{12}x_2 + \cdots + a_{1n}x_n &= b_1 \\
a_{21}x_2 + a_{22}x_2 + \cdots + a_{2n}x_n &= b_2 \\
\cdots\cdots\cdots\cdots\cdots \\
a_{n1}x_n + a_{n2}x_2 + \cdots + a_{nn}x_n &= b_n
\end{aligned}
$$

❖ 矩陣型式

$$\mathbf{Ax} = \mathbf{b}$$

❖ 此處

$$\mathbf{A}=\begin{bmatrix} a_{11} & a_{12}\dots & a_{1n} \\ a_{21} & a_{22}\dots & a_{2n} \\ \vdots & \vdots & \vdots \\ a_{n1} & a_{n2}\dots & a_{nn} \end{bmatrix} \quad \mathbf{b}=\begin{bmatrix} b_1 \\ b_2 \\ \vdots \\ b_n \end{bmatrix} \quad \mathbf{x}=\begin{bmatrix} x_1 \\ x_2 \\ \vdots \\ x_n \end{bmatrix}$$

❖ **A** 稱為係數矩陣

❖ 可得擴大矩陣

$$\mathbf{A}=\begin{bmatrix} a_{11} & a_{12}\dots & a_{1n} & b_1 \\ a_{21} & a_{22}\dots & a_{2n} & b_2 \\ \vdots & \vdots & \vdots & \vdots \\ a_{n1} & a_{n2}\dots & a_{nn} & b_n \end{bmatrix}$$

❖ 基本列運

1. 任兩列互換位置。（亦即相等）

2. 將任一列（亦即相等）乘上非零的純量。

3. 將任一列換以該列與純量乘以該列的和。

2.6 高斯消去法解Ax=b

❖ 應用高斯消去法解下列方程式系統：

$$\begin{bmatrix} 3 & 6 & 9 \\ 2 & (4+p) & 2 \\ -3 & -4 & -11 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ x_3 \end{bmatrix} = \begin{bmatrix} 3 \\ 4 \\ -5 \end{bmatrix} \quad (2.10)$$

❖ 表2.1顯示操作序列

❖ 若p=0 則可得

$$3x_1 + 6x_2 + 9x_3 = 3 \quad (2.11)$$

$$2x_2 - 2x_3 = -2 \quad (2.12)$$

$$-4x_3 = 2 \quad (2.13)$$

表2.1 高斯消去法轉換擴大矩陣為上三角形矩陣

A1	[3]	6	9	3	第一步:初始矩陣
A2	2	(4+p)	2	4	
A3	−3	−4	−11	−5	
A1	3	6	9	3	第二步:第1行之第2列及第3列化爲零
B2＝A2－2(A1)/3	0	p	−4	2	
B3＝A3＋3(A1)/3	0	2	−2	−2	
A1	3	6	9	3	第三步:第2列及第3列互換
B3	0	[2]	−2	−2	
B2	0	p	−4	2	
A1	3	6	9	3	第四步:將第3列第2行化爲零
B3	0	2	−2	−2	
C3＝B2－p(B3)/2	0	0	[(−4+p)]	(2+p)	

❖ 高斯－喬丹消去法(Gauss–Jordan elimination)。

❖ 此方法使用相同的列運算但與高斯消去法不同的是，主對角線上下的元素均為零。

❖ 當p=0 時得到下列擴大矩陣

$$\begin{bmatrix} 3 & 0 & 0 & 16.5 \\ 0 & 2 & 0 & -3.0 \\ 0 & 0 & -4 & 2.0 \end{bmatrix}$$

❖ 所以 $x_1=16.5/3=5.5$、$x_2=-3/2=-1.5$ 和 $x_3=2/-4=-0.5$。

2.7 LU 分解

❖ LU 分解類似高斯消去法，且等效於基本列運算。矩陣A 可以被分解

$$\mathbf{A} = \mathbf{LU} \tag{2.14}$$

❖ **L** 是下三角形矩陣，主對角線為1，**U** 為上三角形矩陣。

❖ 由LU 分解來解方程式系統的主要步驟如下

$$\mathbf{LUx} = \mathbf{b}$$

❖ 令**y=Ux**，則

$$\mathbf{Ly} = \mathbf{b}$$

$$\mathbf{Ux} = \mathbf{y}$$

❖ 以矩陣開始

$$\mathbf{A} = \begin{bmatrix} 3 & 6 & 9 \\ 2 & 5 & 2 \\ -3 & -4 & -11 \end{bmatrix}$$

$\mathbf{U}^{(1)}$ 列2=**A** 的列2−2(**A** 的列1)/3　　(2.15)

$\mathbf{U}^{(1)}$ 列3=**A** 的列3+(3**A** 的列1)/3　　(2.16)

▶數值方法－使用MATLAB程式語言(第三版)

2-1
2-2
2-3
2-4
2-5
2-6
2-7
2-8
2-9
2-10
2-11
2-12
2-13
2-14
2-15
2-16
2-17

❖ **A** 可表示成 $\mathbf{T}^{(1)}\mathbf{U}^{(1)}$ 的乘積如下：

$$\begin{bmatrix} 3 & 6 & 9 \\ 2 & 5 & 2 \\ -3 & -4 & -11 \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 \\ 2/3 & 1 & 0 \\ -1 & 0 & 1 \end{bmatrix} \begin{bmatrix} 3 & 6 & 9 \\ 0 & 1 & -4 \\ 0 & 2 & -2 \end{bmatrix}$$

A 的列2=$\mathbf{U}^{(1)}$ 的列2+2(**A** 的列1)/3　　(2.17)

A 的列2=2($\mathbf{U}^{(1)}$ 的列1)/3+$\mathbf{U}^{(1)}$ 的列2　　(2.18)

❖ $\mathbf{U}^{(1)}$ 變成$\mathbf{T}^{(2)}\mathbf{U}^{(2)}$ 乘積如下：

$$\begin{bmatrix} 3 & 6 & 9 \\ 0 & 1 & -4 \\ 0 & 2 & -2 \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 0 & 1 \\ 0 & 1 & 0 \end{bmatrix} \begin{bmatrix} 3 & 6 & 9 \\ 0 & 2 & -2 \\ 0 & 1 & -4 \end{bmatrix}$$

U的列3=$\mathbf{U}^{(2)}$ 的列3−($\mathbf{U}^{(2)}$ 的列2)/2

❖ 所以$\mathbf{U}^{(2)}$ 變成$\mathbf{T}^{(3)}$U 的乘積如下：

$$\begin{bmatrix} 3 & 6 & 9 \\ 0 & 2 & -2 \\ 0 & 1 & -4 \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 1/2 & 1 \end{bmatrix} \begin{bmatrix} 3 & 6 & 9 \\ 0 & 2 & -2 \\ 0 & 0 & -3 \end{bmatrix}$$

❖ 因此 $\mathbf{A} = \mathbf{T}^{(1)}\ \mathbf{T}^{(2)}\ \mathbf{T}^{(3)}\ \mathbf{U}$ ，意指$\mathbf{L}=\mathbf{T}^{(1)}\mathbf{T}^{(2)}\mathbf{T}^{(3)}$ 如下：

$$\begin{bmatrix} 1 & 0 & 0 \\ 2/3 & 1 & 0 \\ -1 & 0 & 1 \end{bmatrix} \begin{bmatrix} 1 & 0 & 0 \\ 0 & 0 & 1 \\ 0 & 1 & 0 \end{bmatrix} \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 1/2 & 1 \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 \\ 2/3 & 1/2 & 1 \\ -1 & 1 & 0 \end{bmatrix}$$

❖ MATLAB 使用函數lu 完成LU 分解，產生一矩陣但不是一定下三角矩陣。

❖ 以MATLAB 函數lu 處理上例：

```
>> A = [3 6 9;2 5 2;-3 -4 -11]

A =
     3     6     9
     2     5     2
    -3    -4   -11
```



```
>> [L1 U] = lu(A)

L1 =
    1.0000         0         0
    0.6667    0.5000    1.0000
   -1.0000    1.0000         0

U =
     3     6     9
     0     2    -2
     0     0    -3
```

```
>> [L U P] = lu(A)

L =
    1.0000         0         0
   -1.0000    1.0000         0
    0.6667    0.5000    1.0000

U =
     3     6     9
     0     2    -2
     0     0    -3

P =
     1     0     0
     0     0     1
     0     1     0
```


❖ MATLAB 運算元＼用LU 分解求**Ax=b** 的解。

$$\mathbf{A}=\begin{bmatrix}3 & 4 & -5\\ 6 & -3 & 4\\ 8 & 9 & -2\end{bmatrix} \quad \text{及} \quad \mathbf{B}=\begin{bmatrix}1 & 3\\ 9 & 5\\ 9 & 4\end{bmatrix}$$

❖ 完成LU 分解，使**LU=A**，得到

$$\mathbf{L}=\begin{bmatrix}0.375 & -0.064 & 1\\ 0.750 & 1 & 0\\ 1 & 0 & 0\end{bmatrix} \quad \text{及} \quad \mathbf{U}=\begin{bmatrix}8 & 9 & -2\\ 0 & -9.75 & 5.5\\ 0 & 0 & -3.897\end{bmatrix}$$

❖ 所以**LY=B** 由此給出

$$\begin{bmatrix} 0.375 & -0.064 & 1 \\ 0.750 & 1 & 0 \\ 1 & 0 & 0 \end{bmatrix} \begin{bmatrix} y_{11} & y_{12} \\ y_{21} & y_{22} \\ y_{31} & y_{32} \end{bmatrix} = \begin{bmatrix} 1 & 3 \\ 9 & 5 \\ 9 & 4 \end{bmatrix}$$

❖ 這意味著有2 方程式系統，若分開寫則是

$$\mathbf{L}\begin{bmatrix} y_{11} \\ y_{21} \\ y_{31} \end{bmatrix} = \begin{bmatrix} 1 \\ 9 \\ 9 \end{bmatrix} \quad 及 \quad \mathbf{L}\begin{bmatrix} y_{12} \\ y_{22} \\ y_{32} \end{bmatrix} = \begin{bmatrix} 3 \\ 5 \\ 4 \end{bmatrix}$$

▶ 數值方法－使用MATLAB程式語言(第三版)

2-1 2-2 2-3 2-4 2-5 2-6 2-7 2-8 2-9 2-10 2-11 2-12 2-13 2-14 2-15 2-16 2-17

❖ 完全的**Y** 矩陣是

$$\mathbf{Y} = \begin{bmatrix} 9.000 & 4.000 \\ 2.250 & 2.000 \\ -2.231 & 1.628 \end{bmatrix}$$

❖ 最後由後向代入法解UX=Y 得到

$$\mathbf{X} = \begin{bmatrix} 1.165 & 0.891 \\ 0.092 & -0.441 \\ 0.572 & -0.418 \end{bmatrix}$$

2.8 柯列斯基(Cholesky) 分解

❖ 柯列斯基分解是三角分解的一種， 只用於正定對稱(positive definite symmetric) 或正定Hermition 矩陣。

❖ 若A 是對稱或Hermitian 則可寫成

$$\mathbf{A} = \mathbf{P}^{\top}\mathbf{P} \text{ (或 } \mathbf{A} = \mathbf{P}^{H}\mathbf{P}\text{，當A 是Hermitian 時)} \quad (2.19)$$

❖ **P** 的主對角線元素由包括求平方根得出，例如：

$$p_{22} = \sqrt{a_{22} - p_{12}^2}$$

▶ 數值方法－使用MATLAB程式語言（第三版）

❖ 考慮下列正定Hermitian 矩陣的柯列斯基分解：

```
>> A = [2 -i 0;i 2 0;0 0 3]

A =
   2.0000                  0 - 1.0000i          0
        0 + 1.0000i   2.0000                    0
        0                  0               3.0000

>> P = chol(A)

P =
   1.4142                  0 - 0.7071i          0
        0             1.2247                    0
        0                  0               1.7321
```

2-1 2-2 2-3 2-4 2-5 2-6 2-7 2-8 2-9 2-10 2-11 2-12 2-13 2-14 2-15 2-16 2-17

❖ 當運算元＼偵測到一個對稱正定或是Hermitian正定矩陣，則由下列運算解**Ax=b**。

❖ **A** 被分解成 $\mathbf{P}^{\top}\mathbf{P}$，**y** 設定成**Px**，則 $\mathbf{P}^{\top}\mathbf{y}=\mathbf{b}$，因為 $\mathbf{P}^{\top}$ 是下三角矩陣，所以演算法由前向代入法解**y**，**P** 是上三角矩陣所以可由**y** 以後向代入法求**x**。

❖ 例如：

$$\mathbf{A}=\begin{bmatrix}2 & 3 & 4\\ 3 & 6 & 7\\ 4 & 7 & 10\end{bmatrix} \text{ 及 } \mathbf{b}=\begin{bmatrix}2\\ 4\\ 8\end{bmatrix}$$

❖ 由柯列斯基分解得

$$\mathbf{P}=\begin{bmatrix}1.414 & 2.121 & 2.828\\ 0 & 1.225 & 0.817\\ 0 & 0 & 1.155\end{bmatrix}$$

❖ 前向代入法求y 得出

$$\mathbf{y}=\begin{bmatrix}1.414\\ 0.817\\ 2.887\end{bmatrix}$$

❖ 後向代入法解$\mathbf{Px} = \mathbf{y}$得

$$\mathbf{x} = \begin{bmatrix} -2.5 \\ -1.0 \\ 2.5 \end{bmatrix}$$

❖ 下列程式產生對稱正定矩陣，以其轉置矩陣相乘：

```
% e3s205.m
disp('   n         time-backslash   time-chol');
for n = 300:100:1300
    A = [ ]; M = 100*randn(n,n);
    A = M*M'; b = [1:n].';
    tic, x = A\b; t1 = toc;
    tic, R = chol(A);
    v = R.'\b; x = R\b;
    t2 = toc;
    fprintf('%4.0f %14.4f %13.4f \n',n,t1,t2)
end
```

n	time-backslash	time-chol
300	0.0053	0.0073
400	0.0105	0.0115
500	0.0182	0.0216
600	0.0176	0.0197
700	0.0263	0.0281
800	0.0368	0.0385
900	0.0510	0.0519
1000	0.0666	0.0668
1100	0.0862	0.0869
1200	0.1113	0.1065
1300	0.1449	0.1438

2.9 QR 分解

❖ 當 **A** 為實矩陣時分解成上三角與正交矩陣(orthogonal matrix) 的乘積，**A**為複數矩陣時分解成上三角矩陣與單式矩陣(unitary matrix) 的乘積。這稱為QR分解。

$$\mathbf{A} = \mathbf{Q}\,\mathbf{R}$$

❖ 有數種程序提供QR 分解， 這裡提出Householder法(Householder'smethod)，分解一實矩陣，由定義矩陣**P** 開始

$$\mathbf{P} = \mathbf{I} - 2\mathbf{w}\mathbf{w}^{\top} \qquad (2.20)$$

$$\mathbf{PP} = \left(\mathbf{I} - 2\mathbf{w}\mathbf{w}^\top\right)\left(\mathbf{I} - 2\mathbf{w}\mathbf{w}^\top\right)$$
$$= \mathbf{I} - 4\mathbf{w}\mathbf{w}^\top + 4\mathbf{w}\mathbf{w}^\top\left(\mathbf{w}\mathbf{w}^\top\right) = \mathbf{I}$$

$$\mathbf{w}_1^\top = \mu_1[(a_{11} - s_1)\ \ a_{21}\ \ a_{31} \ldots a_{n1}]$$

$$\mu_1 = \frac{1}{\sqrt{2s_1(s_1 - a_{11})}} \quad 及 \quad s_1 = \pm\left(\sum_{j=1}^{n} a_{j1}^2\right)^{1/2}$$

$$\mathbf{A}^{(1)} = \mathbf{P}^{(1)}\mathbf{A} = \begin{bmatrix} s_1 & + & \cdots & + \\ 0 & + & \cdots & + \\ \vdots & \vdots & & \vdots \\ 0 & + & \cdots & + \\ 0 & + & \cdots & + \end{bmatrix}$$

$$\mathbf{w}_2^\top = \mu_2\left[0\ \ \left(a_{22}^{(1)} - s_2\right)\ \ a_{32}^{(1)}\ \ a_{42}^{(1)} \cdots a_{n2}^{(1)}\right]$$

❖ 這裡a_{ij}是$\mathbf{A}$的係數

$$\mu_2 = \frac{1}{\sqrt{2s_2(s_2 - a_{22}^{(1)})}} \quad 及 \quad s_2 = \pm\left(\sum_{j=2}^{n}(a_{j2}^{(1)})^2\right)^{1/2}$$

❖ 正交矩陣 $\mathbf{P}^{(2)}$ 由下式產生

$$\mathbf{P}^{(2)} = \mathbf{I} - 2\mathbf{w}_2\mathbf{w}_2^{\top}$$

❖ 矩陣 $\mathbf{A}^{(2)}$ 由乘積 $\mathbf{P}^{(2)}\mathbf{A}^{(1)}$ 產生如下：

$$\mathbf{A}^{(2)} = \mathbf{P}^{(2)}\mathbf{A}^{(1)} = \mathbf{P}^{(2)}\mathbf{P}^{(1)}\mathbf{A} = \begin{bmatrix} s_1 & + & \cdots & + \\ 0 & s_2 & \cdots & + \\ \vdots & \vdots & & \vdots \\ 0 & 0 & \cdots & + \\ 0 & 0 & \cdots & + \end{bmatrix}$$

▶ 數值方法－使用MATLAB程式語言(第三版)

2-1 2-2 2-3 2-4 2-5 2-6 2-7 2-8 2-9 2-10 2-11 2-12 2-13 2-14 2-15 2-16 2-17

❖ 繼續此步驟 $n-1$ 次直到得到上三角矩陣 $\mathbf{R}$，所以

$$\mathbf{R} = \mathbf{P}^{(n-1)} \ldots \mathbf{P}^{(2)}\mathbf{P}^{(1)}\mathbf{A} \tag{2.21}$$

❖ 由(2.21)

$$\mathbf{Q}^{\top} = \mathbf{P}^{(n-1)} \ldots \mathbf{P}^{(2)}\mathbf{P}^{(1)}$$

❖ 考慮下列矩陣的分解

$$\mathbf{A} = \begin{bmatrix} 4 & -2 & 7 \\ 6 & 2 & -3 \\ 3 & 4 & 4 \end{bmatrix}$$

$$s_1 = \sqrt{\left(4^2 + 6^2 + 3^2\right)} = 7.8102$$

$$\mu_1 = 1/\sqrt{[2 \times 7.8102 \times (7.8102 - 4)]} = 0.1296$$

$$\mathbf{w}_1^{\top} = 0.1296[(4 - 7.8102)\ \ 6\ \ 3] = [-0.4939\ \ 0.7777\ \ 0.3889]$$

▶ 數值方法－使用MATLAB程式語言(第三版)

2-1
2-2
2-3
2-4
2-5
2-6
2-7
2-8
2-9
2-10
2-11
2-12
2-13
2-14
2-15
2-16
2-17

❖ 使用(2.20) 產生$\mathbf{P}^{(1)}$，再產生$\mathbf{A}^{(1)}$，所以

$$\mathbf{P}^{(1)} = \begin{bmatrix} 0.5121 & 0.7682 & 0.3841 \\ 0.7682 & -0.2097 & -0.6049 \\ 0.3841 & -0.6049 & 0.6976 \end{bmatrix}$$

$$\mathbf{A}^{(1)} = \mathbf{P}^{(1)}\mathbf{A} = \begin{bmatrix} 7.8102 & 2.0486 & 2.8168 \\ 0 & -4.3753 & 3.5873 \\ 0 & 0.8123 & 7.2936 \end{bmatrix}$$

❖ 繼續第2 步驟：

$$s_2 = \sqrt{\left\{(-4.3753)^2 + 0.8123^2\right\}} = 4.4501$$

$$\mu_2 = 1/\sqrt{\{2 \times 4.4501 \times (4.4501 + 4.3753)\}} = 0.1128$$

$$\mathbf{w}_2^\top = 0.1128[0 \ \ (-4.3753 - 4.4501) \ \ 0.8123] = [0 \ \ -0.9958 \ \ 0.0917]$$

$$\mathbf{P}^{(2)} = \begin{bmatrix} 1 & 0 & 0 \\ 0 & -0.9832 & 0.1825 \\ 0 & 0.1825 & 0.9832 \end{bmatrix}$$

$$\mathbf{R} = \mathbf{A}^{(2)} = \mathbf{P}^{(2)}\mathbf{A}^{(1)} = \begin{bmatrix} 7.8102 & 2.0486 & 2.8168 \\ 0 & 4.4501 & -2.1956 \\ 0 & 0 & 7.8259 \end{bmatrix}$$

▶ 數值方法－使用MATLAB程式語言(第三版)

2-1
2-2
2-3
2-4
2-5
2-6
2-7
2-8
2-9
2-10
2-11
2-12
2-13
2-14
2-15
2-16
2-17

❖ 最後求正交矩陣**Q** 如下：

$$\mathbf{Q} = \left(\mathbf{P}^{(2)}\mathbf{P}^{(1)}\right)^\top = \begin{bmatrix} 0.5121 & -0.6852 & 0.5179 \\ 0.7682 & 0.0958 & -0.6330 \\ 0.3841 & 0.7220 & 0.5754 \end{bmatrix}$$

❖ 因為MATLAB 提供函數qr 來完成分解。例如：

```
>> A = [4 -2 7;6 2 -3;3 4 4]

A =
        4        -2         7
        6         2        -3
        3         4         4
```

```
>> [Q R] = qr(A)

Q =
   -0.5121    0.6852    0.5179
   -0.7682   -0.0958   -0.6330
   -0.3841   -0.7220    0.5754

R =
   -7.8102   -2.0486   -2.8168
         0   -4.4501    2.1956
         0         0    7.8259
```


2.10 奇異值分解(Singular Value Decomposition；SVD)

❖ $m \times n$ 矩陣A 的奇異值分解如下

$$\mathbf{A} = \mathbf{USV}^{\top} \ (\text{若}\mathbf{A}\text{是複數則為}\mathbf{A}=\mathbf{USV}^{H})$$

❖ **U** 是$m \times m$ 正交矩陣，**V** 是$n \times n$ 正交矩陣。

❖ 在所有情形**S** 是$m \times n$ 實數對角矩陣。

❖ 通常安排為$s_1 > s_2 > s_3 \cdots > s_n$ 降階方式，所以

$$\mathbf{S} = \begin{bmatrix} s_1 & 0 & \dots & 0 \\ 0 & s_2 & \dots & 0 \\ \vdots & \vdots & & \vdots \\ 0 & 0 & \dots & s_n \\ 0 & 0 & \dots & 0 \\ \vdots & \vdots & & \vdots \\ 0 & 0 & \dots & 0 \end{bmatrix}$$

❖ 奇異值是$\mathbf{A}^T\mathbf{A}$ 之特徵值的非負平方根。

❖ 矩陣SVD 有數個重要用途。

- 位階可以由矩陣的SVD 更有效的求出。
- SVD 讓我們檢查病態矩陣的特質。
- SVD 允許我們求矩陣的條件數。

```
>> c = [1 1.01 1.02 1.03 1.04];
>> V = vander(c)

V =
    1.0000    1.0000    1.0000    1.0000    1.0000
    1.0406    1.0303    1.0201    1.0100    1.0000
    1.0824    1.0612    1.0404    1.0200    1.0000
    1.1255    1.0927    1.0609    1.0300    1.0000
    1.1699    1.1249    1.0816    1.0400    1.0000

>> format long
>> s = svd(V)
```



```
s =
   5.210367051037899
   0.101918335876689
   0.000699698839445
   0.000002352380295
   0.000000003294983
```

```
>> norm(V)

ans =
   5.210367051037899

>> cond(V)
```

```
ans =
    1.581303246763933e+009

>> s(1)/s(5)

ans =
    1.581303246763933e+009

>> rank(V)

ans =
     5

>> rref(V)
```



```
ans =
     1     0     0     0     0
     0     1     0     0     0
     0     0     1     0     0
     0     0     0     1     0
     0     0     0     0     1
```

```
>> c = [1 1.01 1.02 1.03 1.03];
>> V = vander(c)

V =
     1.0000     1.0000     1.0000     1.0000     1.0000
     1.0406     1.0303     1.0201     1.0100     1.0000
     1.0824     1.0612     1.0404     1.0200     1.0000
     1.1255     1.0927     1.0609     1.0300     1.0000
     1.1255     1.0927     1.0609     1.0300     1.0000
```


▶ 數值方法－使用MATLAB程式語言(第三版)

```
>> format long e
>> s = svd(V)

s =
     5.187797954424026e+000
     8.336322098941414e-002
     3.997349250042135e-004
     8.462129966456217e-007
                          0

>> format short
>> rank(V)

ans =
      4
```

2-1
2-2
2-3
2-4
2-5
2-6
2-7
2-8
2-9
2-10
2-11
2-12
2-13
2-14
2-15
2-16
2-17

```
>> rref(V)

ans =
    1.0000         0         0         0   -0.9424
         0    1.0000         0         0    3.8262
         0         0    1.0000         0   -5.8251
         0         0         0    1.0000    3.9414
         0         0         0         0         0

>> cond(V)

ans =
   Inf
```


2.11　假反置(pseudo-inverse)

- 若**A** 是mxn 型長方形矩陣，則下列系統

$$\mathbf{Ax} = \mathbf{b} \tag{2.22}$$

- 無法由反置**A** 來求解，因為**A** 並非方陣。
- 假設$m>n$，則以$\mathbf{A}^T$ 預乘(2.22) 可轉系統矩陣為方陣如下：

$$\mathbf{A}^\top\mathbf{Ax} = \mathbf{A}^\top\mathbf{b}$$

- (2.22)的解如下

$$\mathbf{x} = \left(\mathbf{A}^\top\mathbf{A}\right)^{-1}\mathbf{A}^\top\mathbf{b} \tag{2.23}$$

- 令

$$\mathbf{A}^+ = \left(\mathbf{A}^\top\mathbf{A}\right)^{-1}\mathbf{A}^\top \tag{2.24}$$

❖ 矩陣$\mathbf{A}^+$稱為$\mathbf{A}$的假反置，故(2.22)的解是

$$\mathbf{x} = \left(\mathbf{A}^+\right)\mathbf{b} \tag{2.25}$$

❖ 若A是方陣和非奇異的，則$\mathbf{A}^+ = \mathbf{A}^{-1}$。若A是複數則

$$\mathbf{A}^+ = \left(\mathbf{A}^H\mathbf{A}\right)^{-1}\mathbf{A}^H \tag{2.26}$$

其中$\mathbf{A}^H$是轉置共軛

▶數值方法－使用MATLAB程式語言(第三版)

2-1 2-2 2-3 2-4 2-5 2-6 2-7 2-8 2-9 2-10 2-11 2-12 2-13 2-14 2-15 2-16 2-17

❖ 假反置具有下列性質：

1. $\mathbf{A}(\mathbf{A}^+)\mathbf{A}=\mathbf{A}$
2. $(\mathbf{A}^+)\mathbf{A}(\mathbf{A}^+)=\mathbf{A}^+$
3. $(\mathbf{A}^+)\mathbf{A}$ 和$\mathbf{A}(\mathbf{A}^+)$ 是對稱矩陣。

❖ 重寫(2.22) 式

$$\mathbf{Ax} = \left(\mathbf{AA}^\top\right)\left(\mathbf{AA}^\top\right)^{-1}\mathbf{b}$$

$$\mathbf{x} = \mathbf{A}^\top\left(\mathbf{AA}^\top\right)^{-1}\mathbf{b}$$

$$\mathbf{x} = \left(\mathbf{A}^+\right)\mathbf{b}$$

❖ 其中$\mathbf{A}^+=\mathbf{A}^T(\mathbf{AA}^T)^{-1}$ 是假反置。

❖ 若$\mathbf{A}$ 是實數，則$\mathbf{A}$ 的SVD 是$\mathbf{USV}^T$，其中$\mathbf{U}$ 是一正交$m\times m$ 矩陣且$\mathbf{V}$ 是一正交$n\times n$ 矩陣，$\mathbf{S}$ 是一$n\times m$ 的奇異值矩陣。故$\mathbf{A}^T$的SVD 是$\mathbf{VSTU}^T$，所以

$$\mathbf{A}^\top\mathbf{A} = (\mathbf{VS}^\top\mathbf{U}^\top)(\mathbf{USV}^\top) = \mathbf{VS}^\top\mathbf{SV}^\top \text{ 因為 } \mathbf{U}^\top\mathbf{U} = \mathbf{I}$$

$$\begin{aligned}\mathbf{A}^+ &= (\mathbf{VS}^\top\mathbf{SV}^\top)^{-1}\mathbf{VS}^\top\mathbf{U}^\top = \mathbf{V}^{-\top}(\mathbf{S}^\top\mathbf{S})^{-1}\mathbf{V}^{-1}\mathbf{VS}^\top\mathbf{U}^\top \\ &= \mathbf{V}(\mathbf{S}^\top\mathbf{S})^{-1}\mathbf{S}^\top\mathbf{U}^\top \qquad (2.27)\end{aligned}$$

範例2.5

❖ 考慮下列矩陣

$$\mathbf{A} = \begin{bmatrix} 1 & 2 & 3 \\ 4 & 5 & 9 \\ 5 & 6 & 7 \\ -2 & 3 & 1 \end{bmatrix}$$

使用(2.24) 的MATLAB 寫法計算**A** 的假反置可得

▶ 數值方法－使用MATLAB程式語言(第三版)

範例2.5

```
>> A = [1 2 3;4 5 9;5 6 7;-2 3 1];
>> rank(A)

ans =
     3
```

❖ 注意A 是全位階(full rank)，故

```
>> A_cross = inv(A.'*A)*A.'

A_cross =
   -0.0747   -0.1467    0.2500   -0.2057
   -0.0378   -0.2039    0.2500    0.1983
    0.0858    0.2795   -0.2500   -0.0231
```

範例2.5

❖ MATLAB 函數pinv 直接求出這一結果且有較高精確度。

```
A*A_cross*A

ans =
     1.0000    2.0000    3.0000
     4.0000    5.0000    9.0000
     5.0000    6.0000    7.0000
    -2.0000    3.0000    1.0000

>> A*A_cross
```


範例2.5

```
ans =
     0.1070    0.2841    0.0000    0.1218
     0.2841    0.9096    0.0000   -0.0387
     0.0000    0.0000    1.0000   -0.0000
     0.1218   -0.0387   -0.0000    0.9834

>> A_cross*A

ans =
     1.0000    0.0000    0.0000
     0.0000    1.0000    0.0000
    -0.0000   -0.0000    1.0000
```

範例2.6

❖ 考慮下列位階不足矩陣

$$\mathbf{G} = \begin{bmatrix} 1 & 2 & 3 \\ 4 & 5 & 9 \\ 7 & 11 & 18 \\ -2 & 3 & 1 \\ 7 & 1 & 8 \end{bmatrix}$$

▶ 數值方法－使用MATLAB程式語言(第三版)

範例2.6

❖ 使用MATLAB 可得

```
>> G = [1 2 3;4 5 9;7 11 18;-2 3 1;7 1 8]

G =
     1     2     3
     4     5     9
     7    11    18
    -2     3     1
     7     1     8

>> rank(G)

ans =
     2
```

2-1 2-2 2-3 2-4 2-5 2-6 2-7 2-8 2-9 2-10 2-11 2-12 2-13 2-14 2-15 2-16 2-17

範例2.6

2-1 2-2 2-3 2-4 2-5 2-6 2-7 2-8 2-9 2-10 2-11 2-12 2-13 2-14 2-15 2-16 2-17

❖ 注意G 的位階為2，即是位階不足，故無法由(2.24) 式求其假反置。現在求G 的SVD 如下：

```
>> [U S V] = svd(G)

U =
   -0.1381    0.0839    0.9724   -0.0044   -0.1681
   -0.4115    0.0215    0.0539   -0.6081    0.6764
   -0.8258    0.2732   -0.2165    0.0607   -0.4392
   -0.0524    0.5650    0.0366    0.6373    0.5201
   -0.3563   -0.7737    0.0572    0.4695    0.2253
```


範例2.6

2-1 2-2 2-3 2-4 2-5 2-6 2-7 2-8 2-9 2-10 2-11 2-12 2-13 2-14 2-15 2-16 2-17

```
S =
   26.8394         0         0
         0    6.1358         0
         0         0    0.0000
         0         0         0
         0         0         0
```

```
V =
   -0.3709   -0.7274   -0.5774
   -0.4445    0.6849   -0.5774
   -0.8154   -0.0425    0.5774
```

❖ 現在選擇兩個重要的奇異值作隨後計算之用：

範例2.6

```
>> SS = S(1:2,1:2)

SS =
   26.8394         0
         0    6.1358
```

❖ 為了讓矩陣乘法是共容的，我們只取U 和V 的前 2 行如下：

```
>> G_cross = V(:,1:2)*inv(SS.'*SS)*SS.'*U(:,1:2).'

G_cross =
   -0.0080    0.0031   -0.0210   -0.0663    0.0966
    0.0117    0.0092    0.0442    0.0639   -0.0805
    0.0036    0.0124    0.0232   -0.0023    0.0162
```


▶ 數值方法－使用MATLAB程式語言(第三版)

範例2.6

❖ 這結果可直接由pinv 函數直接求得，其是基於A 的奇異值分解。

```
>> G*G_cross

ans =
    0.0261    0.0586    0.1369    0.0546   -0.0157
    0.0586    0.1698    0.3457    0.0337    0.1300
    0.1369    0.3457    0.7565    0.1977    0.0829
    0.0546    0.0337    0.1977    0.3220   -0.4185
   -0.0157    0.1300    0.0829   -0.4185    0.7256

>> G_cross*G
```

2-1 2-2 2-3 2-4 2-5 2-6 2-7 2-8 2-9 2-10 2-11 2-12 2-13 2-14 2-15 2-16 2-17

範例2.6

```
ans =
     0.6667    -0.3333     0.3333
    -0.3333     0.6667     0.3333
     0.3333     0.3333     0.6667
```

❖ 注意 G*G_cross 及 G_cross*G 兩者皆是對稱的。

2.12 過定與欠定系統

❖ 考慮以下高於所求型線性方程式系統：

$$\begin{aligned} x_1 + x_2 &= 1.98 \\ 2.05x_1 - x_2 &= 0.95 \\ 3.06x_1 + x_2 &= 3.98 \\ -1.02x_1 + 2x_2 &= 0.92 \\ 4.08x_1 - x_2 &= 2.90 \end{aligned} \tag{2.28}$$

❖ 圖2.5 顯示(2.28) 就是高於所求型系統，直線方程式並不相交於一點。

2-1 2-2 2-3 2-4 2-5 2-6 2-7 2-8 2-9 2-10 2-11 2-12 2-13 2-14 2-15 2-16 2-17

❖ 考慮方程式系統(2.28)，令$r_1, \cdots, r_5$ 是殘值餘數，則

$$\begin{aligned} x_1 + x_2 - 1.98 &= r_1 \\ 2.05x_1 - x_2 - 0.95 &= r_2 \\ 3.06x_1 + x_2 - 3.98 &= r_3 \\ -1.02x_1 + 2x_2 - 0.92 &= r_4 \\ 4.08x_1 - x_2 - 2.90 &= r_5 \end{aligned}$$

2-1 2-2 2-3 2-4 2-5 2-6 2-7 2-8 2-9 2-10 2-11 2-12 2-13 2-14 2-15 2-16 2-17

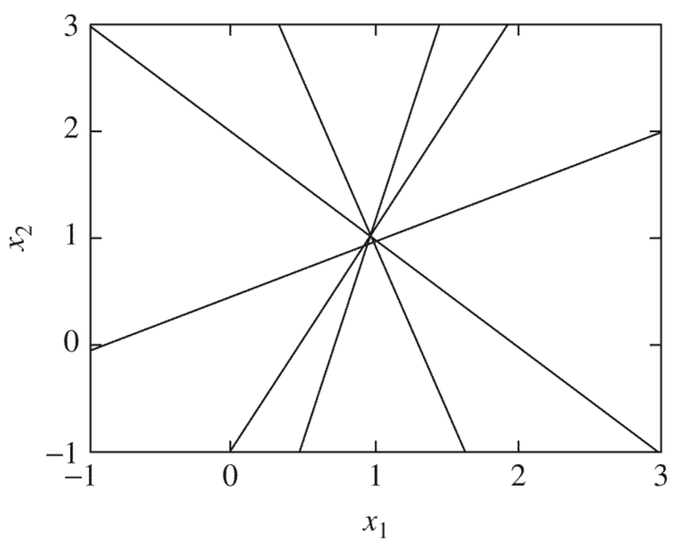

圖2.5　方程式系統(2.28) 的不一致性(inconsistents) 繪圖。

❖ 殘數平方和給定為

$$S=\sum_{i=1}^{5} r_i^2 \tag{2.29}$$

❖ 欲將S 最小化可經由

$$\frac{\partial S}{\partial x_k}=0,\quad k=1,2$$

$$\frac{\partial S}{\partial x_k}=\sum_{i=1}^{5} 2r_i\frac{\partial r_i}{\partial x_k},\quad k=1,2$$

$$\sum_{i=1}^{5} r_i\frac{\partial r_i}{\partial x_k}=0,\quad k=1,2 \tag{2.30}$$

❖ 比較解(2.28) 式時，使用運算子＼及假反置法所得的結果。MATLAB程式為

```
% e3s206.m
A = [1 1;2.05 -1;3.06 1;-1.02 2;4.08 -1];
b = [1.98;0.95;3.98;0.92;2.90];
x = pinv(A)*b
norm_pinv = norm(A*x-b)
x = A\b
norm_op = norm(A*x-b)
```

```
x =
     0.9631
     0.9885
```

```
norm_pinv =
     0.1064

x =
     0.9631
     0.9885

norm_op =
     0.1064
```

2-1 2-2 2-3 2-4 2-5 2-6 2-7 2-8 2-9 2-10 2-11 2-12 2-13 2-14 2-15 2-16 2-17

❖ 圖2.6 比圖2.5 更仔細的顯示這些方程式交點的情形。

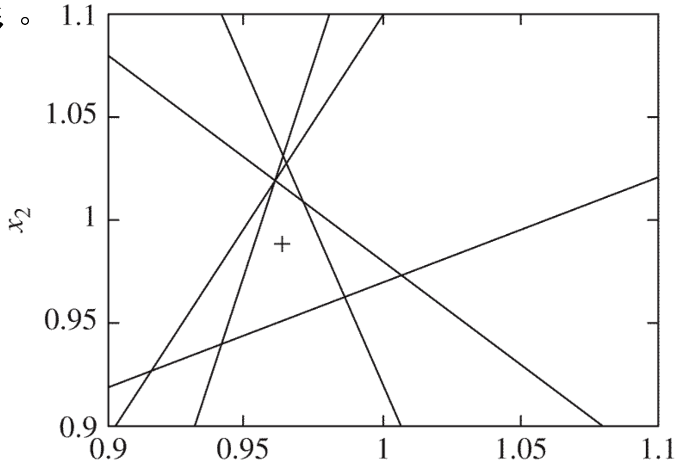

圖2.6 (2.28) 式不相容方程式系統的繪圖，顯示直線方程式的交點區，其中“+”表示“最佳”解。

2-1 2-2 2-3 2-4 2-5 2-6 2-7 2-8 2-9 2-10 2-11 2-12 2-13 2-14 2-15 2-16 2-17

❖ 應用MATLAB 函數qr 解(2.28) 式之高於所求型系統可得：

```
>> A = [1 1;2.05 -1;3.06 1;-1.02 2;4.08 -1];
>> b = [1.98 0.95 3.98 0.92 2.90].';
>> [Q R] = qr(A)

Q =
   -0.1761    0.4123   -0.7157   -0.2339   -0.4818
   -0.3610   -0.2702    0.0998    0.6751   -0.5753
   -0.5388    0.5083    0.5991   -0.2780   -0.1230
    0.1796    0.6839   -0.0615    0.6363    0.3021
   -0.7184   -0.1756   -0.3394    0.0857    0.5749

R =
   -5.6792    0.7237
         0    2.7343
         0         0
         0         0
         0         0
```


▶ 數值方法－使用MATLAB程式語言(第三版)

❖ 繼續上面的例子

```
>> y = Q.'*b

y =
   -4.7542
    2.7029
    0.0212
   -0.0942
   -0.0446
```

$$-5.6792x_1 + 0.7237x_2 = -4.7542$$

$$2.7343x_2 = 2.7029$$

2-1 2-2 2-3 2-4 2-5 2-6 2-7 2-8 2-9 2-10 2-11 2-12 2-13 2-14 2-15 2-16 2-17

❖ 求出 $x_1 = 0.9631$ 和 $x_2 = 0.9885$。

❖ 以下例子是位階不足且表示平行線系統。

$$x_1 + 2x_2 = 1.00$$
$$x_1 + 2x_2 = 1.03$$
$$x_1 + 2x_2 = 0.97$$
$$x_1 + 2x_2 = 1.01$$


```
>> A = [1 2;1 2;1 2;1 2]

A =
     1     2
     1     2
     1     2
     1     2

>> b = [1 1.03 0.97 1.01].'

b =
    1.0000
    1.0300
    0.9700
    1.0100

>> y = A\b
Warning: Rank deficient, rank = 1,  tol =   3.552714e-015.
```

```
y =
         0
    0.5012

>> norm(y)

ans =
    0.5012
```


❖ pinv 函數求解如下：

```
>> x = pinv(A)*b

x =
    0.2005
    0.4010

>> norm(x)

ans =
    0.4483
```

❖ 可見當函數pinv 及＼運算元應用至位階不足系統時，pinv 函數得出的解有最小的歐基里德範數。

❖ 考慮方程式系統

$$x_1 + 2x_2 + 3x_3 + 4x_4 = 1$$
$$-5x_1 + 3x_2 + 2x_3 + 7x_4 = 2$$


```
>> A = [1 2 3 4;-5 3 2 7];
>> b = [1 2].';
>> x1 = A\b

x1 =
   -0.0370
         0
         0
    0.2593

>> x2 = pinv(A)*b

x2 =
   -0.0780
    0.0787
    0.0729
    0.1755
```

```
>> norm(x1)

ans =
     0.2619

>> norm(x2)

ans =
     0.2199
```

❖ 第一個解x1 是滿足系統的一個解，第二個解x2 也滿足方程式系統但同時給出最小範數之解。

▶數值方法－使用MATLAB程式語言(第三版)

2-1
2-2
2-3
2-4
2-5
2-6
2-7
2-8
2-9
2-10
2-11
2-12
2-13
2-14
2-15
2-16
2-17

❖ 考慮以lsqnonneg 函數解非負最小平方問題。

❖ 方程式為

$$\mathbf{Ax} = \mathbf{b} \quad \text{受限於} \quad \mathbf{x} \geq \mathbf{0}$$

❖ 其中**A** 及**b** 必須是實數。這問題等效於求向量**x**。向量**x** 最小化norm(**Ax**−**b**)，受限於**x**≥**0**。

❖ 呼叫MATLAB 函數lsqnonneg

```
x = lsqnonneg(A,b)
```

❖ 考慮下列問題，解

$$\begin{bmatrix} 1 & 1 & 1 & 1 & 0 \\ 1 & 2 & 3 & 0 & 1 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ x_3 \\ x_4 \\ x_5 \end{bmatrix} = \begin{bmatrix} 7 \\ 12 \end{bmatrix}$$

❖ 受制於$x_i \geq 0, i=1,2,\cdots,5$。在MATLAB 內這問題變成

```
>> A = [1 1 1 1 0;1 2 3 0 1];
>> b = [7 12].';
```



```
>> x = lsqnonneg(A,b)

x =
     0
     0
     4
     3
     0
```

❖ 可以使用＼求解但不確保x 是非負值。

```
>> x2 = A\b

x2 =
         0
         0
    4.0000
    3.0000
         0
```

❖ 以下例子說明lsqnonneg 函數如何令非負值解最佳的滿足方程式：

$$\begin{bmatrix} 3.0501 & 4.8913 \\ 3.2311 & -3.2379 \\ 1.6068 & 7.4565 \\ 2.4860 & -0.9815 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} = \begin{bmatrix} 2.5 \\ 2.5 \\ 0.5 \\ 2.5 \end{bmatrix} \qquad (2.31)$$

```
>> A = [3.0501 4.8913;3.2311 -3.2379; 1.6068 7.4565;2.4860 -0.9815];
>> b = [2.5 2.5 0.5 2.5].';
```

❖ 可使用＼或lsqnonneg 函數計算求解：

▶ 數值方法－使用MATLAB程式語言(第三版)

```
>> x1 = A\b

x1 =
    0.8307
   -0.0684
```

```
>> x2 = lsqnonneg(A,b)

x2 =
    0.7971
         0

>> norm(A*x1-b)

ans =
    0.7040

>> norm(A*x2-b)

ans =
    0.9428
```

2-1
2-2
2-3
2-4
2-5
2-6
2-7
2-8
2-9
2-10
2-11
2-12
2-13
2-14
2-15
2-16
2-17

2.13 迭代法

❖ 函數的迭代解發展如下，以線性方程式系統開始

$$\begin{array}{lllll} a_{11}x_1+ & a_{12}x_2+ & \ldots & +a_{1n}x_n & =b_1 \\ a_{21}x_1+ & a_{22}x_2+ & \ldots & +a_{2n}x_n & =b_2 \\ \vdots & \vdots & & \vdots & \vdots \\ a_{n1}x_1+ & a_{n2}x_2+ & \ldots & +a_{nn}x_n & =b_n \end{array}$$

❖ 重新排列成

▶數值方法－使用MATLAB程式語言(第三版)

$$\begin{array}{l} x_1=(b_1-a_{12}x_2-a_{13}x_3-\ldots-a_{1n}x_n)\,/\,a_{11} \\ x_2=(b_2-a_{21}x_1-a_{23}x_3-\ldots-a_{2n}x_n\,/\,a_{22} \\ \quad\vdots \qquad \vdots \qquad\qquad \vdots \qquad \vdots \\ x_n=(b_n-a_{n1}x_1-a_{n2}x_2-\ldots-a_{n,n-1}x_{n-1})\,/\,a_{nn} \end{array}$$

❖ 如果假設*xi* 初值，其中i=1,…,n，然後代入上式右側則由(2.32) 式可得出新的x_i值，迭代解由這些x_i代入方程式右側……等等。

❖ 這稱為Jacobi 或同時迭代(Simultaneous iteration)。

2-1 2-2 2-3 2-4 2-5 2-6 2-7 2-8 2-9 2-10 2-11 2-12 2-13 2-14 2-15 2-16 2-17

❖ 一旦由(2.32) 式第一方程式求出新的x_1 值，則與舊的$x_3,\cdots,x_n$ 值用在第二方程式求出x_2。這稱為Gauss−Seidel 或循環迭代。

❖ 收斂條件為

$$|a_{ii}| >> \sum_{j=1,\ j\neq i}^{n} |a_{ij}| \qquad i = 1, 2, \ldots, n$$

▶數值方法－使用MATLAB程式語言(第三版)

2.14 稀疏矩陣

2-1
2-2
2-3
2-4
2-5
2-6
2-7
2-8
2-9
2-10
2-11
2-12
2-13
2-14
2-15
2-16
2-17

❖ 稀疏矩陣出現在很多科學及工程問題。

❖ 一矩陣如果包含大量的零元素稱為稀疏矩陣。

❖ 如果用者視一矩陣為稀疏且想使用這優點則首先須轉成稀疏形式。這由函數sparse 達成。

❖ b=sparse(a) 將矩陣a 轉成稀疏矩陣，隨後的MATLAB 運算就會計算這稀疏性。

❖ 若想要將矩陣還原成滿的，則只需使用c = full(b)。

❖ 下列情況說明其用法

```
>> colpos = [1 2 1 2 5 3 4 3 4 5];
>> rowpos = [1 1 2 2 2 4 4 5 5 5];
>> value = [12 -4 7 3 -8 -13 11 2 7 -4];
>> A = sparse(rowpos,colpos,value,5,5)
```

```
A =
   (1,1)        12
   (2,1)         7
   (1,2)        -4
   (2,2)         3
```


2-1
2-2
2-3
2-4
2-5
2-6
2-7
2-8
2-9
2-10
2-11
2-12
2-13
2-14
2-15
2-16
2-17

```
(4,3)       -13
(5,3)         2
(4,4)        11
(5,4)         7
(2,5)        -8
(5,5)        -4
```

❖ 稀疏矩陣轉成滿形矩陣如下：

```
>> B = full(A)

B =
    12    -4     0     0     0
     7     3     0     0    -8
     0     0     0     0     0
     0     0   -13    11     0
     0     0     2     7    -4
```

```
>> [issparse(A) issparse(B) nnz(A) nnz(B)]

ans =
       1      0     10     10
```

❖ 下一例子說明如何產生大的5000×5000 稀疏矩陣，程式如下：

```
% e3s207.m Generates a sparse triple diagonal matrix
n = 5000;
rowpos = 2:n; colpos = 1:n-1;
values = 2*ones(1,n-1);
Offdiag = sparse(rowpos,colpos,values,n,n);
A = sparse(1:n,1:n,4*ones(1,n),n,n);
A = A+Offdiag+Offdiag.';
%generate full matrix
B = full(A);
```



```
%generate arbitrary right hand side for system of equations
rhs = [1:n].';
tic, x = A\rhs; f1 = toc;
tic, x = B\rhs; f2 = toc;
fprintf('Time to solve sparse matrix = %8.5f\n',f1);
fprintf('Time to solve  full  matrix = %8.5f\n',f2);
```

```
Time to solve sparse matrix =  0.00051
Time to solve  full  matrix =  5.74781
```

❖ 相同問題以lu分解5000x5000 矩陣如下：

```
% e3s208.m
n = 5000;
offdiag = sparse(2:n,1:n-1,2*ones(1,n-1),n,n);
A = sparse(1:n,1:n,4*ones(1,n),n,n);
A = A+offdiag+offdiag';
%generate full matrix
B = full(A);
%generate arbitrary right hand side for system of equations
rhs = [1:n]';
tic, lu1 = lu(A); f1 = toc;
tic, lu2 = lu(B); f2 = toc;
fprintf('Time for sparse LU = %8.4f\n',f1);
fprintf('Time for  full  LU = %8.4f\n',f2);
```


▸數值方法－使用MATLAB程式語言(第三版)

2-1
2-2
2-3
2-4
2-5
2-6
2-7
2-8
2-9
2-10
2-11
2-12
2-13
2-14
2-15
2-16
2-17

❖ 所需的浮點運算次數是

```
Time for sparse LU =    0.0056
Time for  full  LU =    9.6355
```

❖ 產生稀疏矩陣的方式是使用函數sprandn 及 sprandsym

```
A = sprandn(m,n,d)
```

❖ 產生*m*x*n* 亂數矩陣，非零元素是常態分佈，密度為d，密度是非零元素數目與全部元素數目之比。

```
A = sprandsys(n,d)
```

❖ 例子如下：

```
>> A = sprandn(5,5,0.25)

A =
   (2,1)      -0.4326
   (3,3)      -1.6656
   (5,3)      -1.1465
   (4,4)       0.1253
   (5,4)       1.1909
   (4,5)       0.2877

>> B = full(A)
```



```
B =
         0         0         0         0         0
   -0.4326         0         0         0         0
         0         0   -1.6656         0         0
         0         0         0    0.1253    0.2877
         0         0   -1.1465    1.1909         0

>> As = sprandsym(5,0.25)

As =
   (3,1)       0.3273
   (1,3)       0.3273
   (5,3)       0.1746
   (5,4)      -0.0376
   (3,5)       0.1746
   (4,5)      -0.0376
   (5,5)       1.1892

>> Bs = full(As)
```

```
Bs =
         0         0    0.3273         0         0
         0         0         0         0         0
    0.3273         0         0         0    0.1746
         0         0         0         0   -0.0376
         0         0    0.1746   -0.0376    1.1892
```


❖ sprandsym 另一種呼叫如下：

```
A = sprandsym(n,density,r)
```

❖ 呼叫型式如下：

```
>> Apd = sprandsym(6,0.4,[1 2.5 6 9 2 4.3])

Apd =
   (1,1)        1.0058
   (2,1)       -0.0294
   (4,1)       -0.0879
   (1,2)       -0.0294
   (2,2)        8.3477
   (4,2)       -1.9540
   (3,3)        5.4937
   (5,3)       -1.3300
   (1,4)       -0.0879
   (2,4)       -1.9540
   (4,4)        3.1465
   (3,5)       -1.3300
   (5,5)        2.5063
   (6,6)        4.3000
```

```
>> Bpd = full(Apd)

Bpd =
    1.0058   -0.0294         0   -0.0879         0         0
   -0.0294    8.3477         0   -1.9540         0         0
         0         0    5.4937         0   -1.3300         0
   -0.0879   -1.9540         0    3.1465         0         0
         0         0   -1.3300         0    2.5063         0
         0         0         0         0         0    4.3000
```


▶ 數值方法－使用MATLAB程式語言(第三版)

2-1
2-2
2-3
2-4
2-5
2-6
2-7
2-8
2-9
2-10
2-11
2-12
2-13
2-14
2-15
2-16
2-17

❖ 可以使用spy 函數檢查預排程序，函數symand 在MATLAB 裡完成對稱最低階數排序(*symmetric minimum degree ordering*)。

❖ 例如：

```
% e3s209.m
% generate a sparse matrix
n = 3000;
offdiag = sparse(2:n,1:n-1,2*ones(1,n-1),n,n);
offdiag2 = sparse(4:n,1:n-3,3*ones(1,n-3),n,n);
offdiag3 = sparse(n-5:n,1:6,7*ones(1,6),n,n);
A = sparse(1:n,1:n,4*ones(1,n),n,n);
A = A+offdiag+offdiag'+offdiag2+offdiag2'+offdiag3+offdiag3';
A = A*A.';
```

```
% generate full matrix
B = full(A);
m_order = symamd(A);
tic
spmult = A(m_order,m_order)*A(m_order,m_order).';
flsp = toc;
tic, fulmult = B*B.'; flful = toc;
fprintf('Time for sparse mult = %6.4f\n',flsp)
fprintf('Time for  full  mult = %6.4f\n',flful)
```

```
Time for sparse mult = 0.0184
Time for  full  mult = 3.8359
```


▶ 數值方法－使用MATLAB程式語言(第三版)

2-1
2-2
2-3
2-4
2-5
2-6
2-7
2-8
2-9
2-10
2-11
2-12
2-13
2-14
2-15
2-16
2-17

❖ 下列程式檢查LU分解。

```
% e3s210.m
% generate a sparse matrix
n = 100;
offdiag = sparse(2:n,1:n-1,2*ones(1,n-1),n,n);
offdiag2 = sparse(4:n,1:n-3,3*ones(1,n-3),n,n);
offdiag3 = sparse(n-5:n,1:6,7*ones(1,6),n,n);
A = sparse(1:n,1:n,4*ones(1,n),n,n);
A = A+offdiag+offdiag'+offdiag2+offdiag2'+offdiag3+offdiag3';
A = A*A.';
A1 = flipud(A);
A = A+A1;
n1 = nnz(A)
B = full(A); %generate full matrix
m_order = symamd(A);
```

```
tic, lud = lu(A(m_order,m_order)); flsp = toc;
n2 = nnz(lud)
tic, fullu = lu(B); flful = toc;
n3 = nnz(fullu)
subplot(2,2,1), spy(A,'k');
title('Original matrix')
subplot(2,2,2), spy(A(m_order,m_order),'k')
title('Ordered matrix')
subplot(2,2,3), spy(fullu,'k')
title('LU decomposition,unordered matrix')
subplot(2,2,4), spy(lud,'k')
title('LU decomposition, ordered matrix')
fprintf('Time for sparse lu = %6.4f\n',flsp)
fprintf('Time for  full  lu = %6.4f\n',flful)
```


▶數值方法－使用MATLAB程式語言(第三版)

2-1
2-2
2-3
2-4
2-5
2-6
2-7
2-8
2-9
2-10
2-11
2-12
2-13
2-14
2-15
2-16
2-17

```
n1 =
         2096

n2 =
         1307
```

```
n3 =
         4465

Time for sparse lu = 0.0013
Time for  full  lu = 0.0047
```

2-1 2-2 2-3 2-4 2-5 2-6 2-7 2-8 2-9 2-10 2-11 2-12 2-13 2-14 2-15 2-16 2-17

❖若矩陣的大小從100×100 增加到3000×3000，則從程式稿的輸出是

```
n1 =
       65896

n2 =
       34657
```

```
n3 =
      526810

Time for sparse lu = 0.0708
Time for  full   lu = 2.3564
```


▸數值方法－使用MATLAB程式語言(第三版)

2-1 2-2 2-3 2-4 2-5 2-6 2-7 2-8 2-9 2-10 2-11 2-12 2-13 2-14 2-15 2-16 2-17

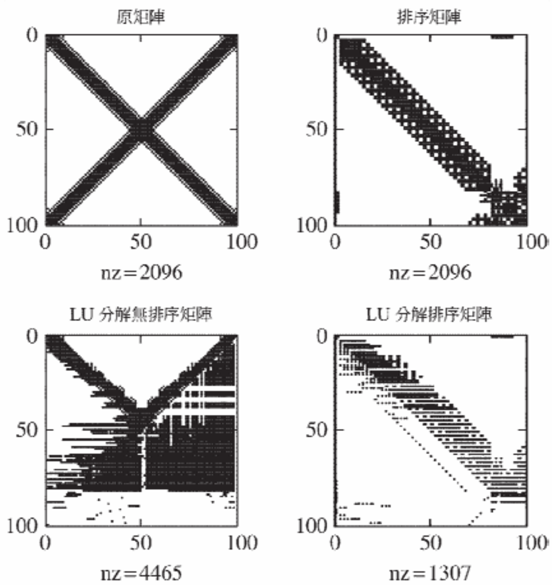

圖2.7　在LU 分解上，最低階數排序的影響。

- MATLAB 函數symand 提供對稱矩陣的最低階數排序。對於非對稱矩陣MATLAB 提供函數colmmd，得到非對稱矩陣的行最低階排序。
- 一般採用稀疏性計算可以節省浮點運算次數，如下列程式說明。

▸數値方法－使用MATLAB程式語言(第三版)

2-1
2-2
2-3
2-4
2-5
2-6
2-7
2-8
2-9
2-10
2-11
2-12
2-13
2-14
2-15
2-16
2-17

```
% e3s211.m
n = 1000;   b = 1:n;
disp('   density    time_sparse    time_full');
for density = 0.004:0.003:0.039
    A = sprandsym(n,density)+0.1*speye(n);
    density = density+1/n;
    tic, x = A\b'; f1 = toc;
    B = full(A);
    tic, y = B\b'; f2 = toc;
    fprintf('%10.4f %12.4f %12.4f\n',density,f1,f2);
end
```

❖ 如果原$n \times n$ 矩陣密度為d，則假設原來對角線是零產生的修改密度為$d+1/n$。

density	time_sparse	time_full
0.0050	0.0204	0.1907
0.0080	0.0329	0.1318
0.0110	0.0508	0.1332
0.0140	0.0744	0.1399
0.0170	0.0892	0.1351
0.0200	0.1064	0.1372
0.0230	0.1179	0.1348
0.0260	0.1317	0.1381
0.0290	0.1444	0.1372
0.0320	0.1516	0.1369
0.0350	0.1789	0.1404
0.0380	0.1627	0.1450

❖ 稀疏矩陣另一重要應用是解最小平方(least squares)問題。

❖ 使用下列程式：

```
% e3s212.m
% generate a sparse triple diagonal matrix
n = 1000;
rowpos = 2:n;  colpos = 1:n-1;
values = ones(1,n-1);
offdiag = sparse(rowpos,colpos,values,n,n);
A = sparse(1:n,1:n,4*ones(1,n),n,n);
A = A+offdiag+offdiag';
%Now generate a sparse least squares system
Als = A(:,1:n/2);
%generate full matrix
Cfl = full(Als);
```

```
tic, x = Als\rhs'; f1 = toc;
tic, x = Cfl\rhs'; f2 = toc;
fprintf('Time for sparse least squares solve = %8.4f\n',f1)
fprintf('Time for  full  least squares solve = %8.4f\n',f2)
```

```
Time for sparse least squares solve =   0.0023
Time for  full  least squares solve =   0.2734
```


2.15 特徵值問題

❖ 研究圖2.8 所示的質量和彈簧系統的特例。這系統的移動方程式為

$$\begin{aligned} m_1\ddot{q}_1 + (k_1+k_2+k_4)\,q_1 - k_2q_2 - k_4q_3 &= 0 \\ m_2\ddot{q}_2 - k_2q_1 + (k_2+k_3)\,q_2 - k_3q_3 &= 0 \\ m_3\ddot{q}_3 - k_4q_1 - k_3q_2 + (k_3+k_4)\,q_3 &= 0 \end{aligned} \quad (2.33)$$

❖ m_1，m_2，m_3 是系統質量，$k_1,\cdots,k_4$ 是彈簧黏滯性(stiffnesses)。

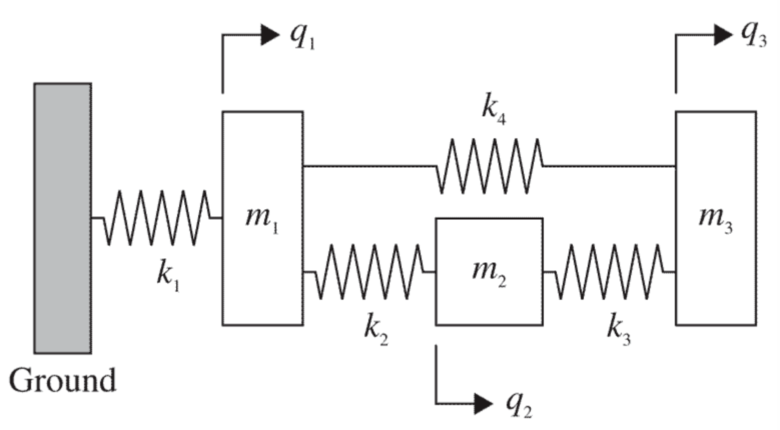

圖2.8　三個自由度的質量-彈簧系統。

▶數值方法－使用MATLAB程式語言(第三版)

❖ $d^2q_i/d^2t=-\omega^2u_i$ $\exp(j\omega t)$，代入(2.33) 且消去共同項 $\exp(j\omega t)$ 得到

$$
\begin{aligned}
-\omega^2 m_1 u_1 + (k_1+k_2+k_4)\,u_1 - k_2u_2 - k_4u_3 &= 0\\
-\omega^2 m_2 u_2 - k_2u_1 + (k_2+k_3)\,u_2 - k_3u_3 &= 0\\
-\omega^2 m_3 u_3 - k_4u_1 - k_3u_2 + (k_3+k_4)\,u_3 &= 0
\end{aligned}
\qquad (2.34)
$$

❖ 若 m_1=10kg，m_2=20kg，m_3=30kg，k_1=10kN/m，k_2=20kN/m，k_3=25kN/m 和 k_4=15kN/m，則(2.34)變成

2-1 2-2 2-3 2-4 2-5 2-6 2-7 2-8 2-9 2-10 2-11 2-12 2-13 2-14 2-15 2-16 2-17

$$-\omega^2 10u_1 + 45000u_1 - 20000u_2 - 15000u_3 = 0$$
$$-\omega^2 20u_1 - 20000u_1 + 45000u_2 - 25000u_3 = 0$$
$$-\omega^2 30u_1 - 15000u_1 - 25000u_2 + 40000u_3 = 0$$

$$-\omega^2 \mathbf{Mu} + \mathbf{Ku} = \mathbf{0} \qquad (2.35)$$

$$\mathbf{M} = \begin{bmatrix} 10 & 0 & 0 \\ 0 & 20 & 0 \\ 0 & 0 & 30 \end{bmatrix} \text{kg} \quad 及 \quad \mathbf{K} = \begin{bmatrix} 45 & -20 & -15 \\ -20 & 45 & -25 \\ -15 & -25 & 40 \end{bmatrix} \text{kN/m}$$

▶ 數值方法－使用MATLAB程式語言(第三版)

❖ 方程式(2.35) 可以用不同方法重排，例如可改寫成

$$\mathbf{Mu} = \lambda \mathbf{Ku} \quad 其中 \quad \lambda = \frac{1}{\omega^2} \qquad (2.36)$$

❖ MATLAB 可用提供函數eig 解特徵值問題，應用它來解(2.35) 以說明其用途。

```
>> M = [10 0 0;0 20 0;0 0 30];
>> K = 1000*[45 -20 -15;-20 45 -25;-15 -25 40];
>> lambda = eig(M,K).'

lambda =
    0.0002    0.0004    0.0073

>> omega = sqrt(1./lambda)

omega =
   72.2165   52.2551   11.7268
```


- 圖2.8 之振動自然頻率是11.72、52.25 及72.21 弳度／秒。
- 思考這一問題的標準形式：

$$\mathbf{A}\mathbf{x} = \lambda \mathbf{x} \qquad (2.37)$$

- 方程式(2.37) 可寫成

$$(\mathbf{A} - \lambda \mathbf{I})\mathbf{x} = \mathbf{0} \qquad (2.38)$$

- 滿足(2.38) 的值是下列方程式的根

$$|\mathbf{A} - \lambda \mathbf{I}| = 0 \qquad (2.39)$$

❖ 例如，

$$\mathbf{A} = \begin{bmatrix} 1 & 2 & 3 \\ 4 & 5 & -6 \\ 7 & -8 & 9 \end{bmatrix}$$

```
>> A = [1 2 3;4 5 -6;7 -8 9];
>> p = poly(A)

p =
     1.0000   -15.0000   -18.0000   360.0000
```


▶ 數值方法－使用MATLAB程式語言(第三版)

2-1
2-2
2-3
2-4
2-5
2-6
2-7
2-8
2-9
2-10
2-11
2-12
2-13
2-14
2-15
2-16
2-17

```
>> roots(p).'

ans =
   14.5343   -4.7494    5.2152
```

```
>> eig(A).'

ans =
   -4.7494    5.2152   14.5343
```

❖ 得到這些值後代回(2.38) 式得到特徵向量的線性方程式：

$$(\mathbf{A}-\lambda_i\mathbf{I})\mathbf{x}=\mathbf{0}\quad i=1,2,\ldots,n \tag{2.40}$$

❖ 若$\lambda_i,\mathbf{x}_i$ 及$\lambda_j,\mathbf{x}_j$ 滿足特徵值問題(2.37) 且λ_i, λ_j 是互異之特徵值，則

$$\lambda_i,\mathbf{x}_i \quad 及 \quad \lambda_j,\mathbf{x}_j \tag{2.41}$$

$$\mathbf{x}_i^\top\mathbf{x}_j=0\quad i\neq j \tag{2.42}$$

❖ 若調整任意的純量乘數以致於

$$\mathbf{x}_i^\top\mathbf{x}_i=1 \tag{2.43}$$

$$\mathbf{x}_i^\top\mathbf{A}\mathbf{x}_i=\lambda_i \tag{2.44}$$

❖ 特徵向量稱為規一化(normalized)。

❖ 若是λ_i=λ_j 與其它特徵值λ_k 互異，則

$$\left.\begin{matrix}\mathbf{x}_i^\top\mathbf{x}_k=0\\ \mathbf{x}_j^\top\mathbf{x}_k=0\end{matrix}\right\}\quad k=1,2,\ldots,n,\quad k\neq i,\quad k\neq j \tag{2.45}$$

❖ 考慮**A** 是實數但不對稱，一對相關的特徵值問題出現如下：

$$\mathbf{Ax} = \lambda \mathbf{x} \tag{2.46}$$

$$\mathbf{A}^\top \mathbf{y} = \beta \mathbf{y} \tag{2.47}$$

❖ (2.47) 式可轉置成

$$\mathbf{y}^\top \mathbf{A} = \beta \mathbf{y}^\top \tag{2.48}$$

❖ 若 $\lambda_i, \mathbf{x}_i, \mathbf{y}_i$ 及 $\lambda_j, \mathbf{x}_j, \mathbf{y}_j$ 是滿足(2.46) 及(2.47)式的解且λ_i 和λ_j 互異，則

$$\mathbf{x}_i^\top \mathbf{x}_j = 0 \quad i \neq j \tag{2.49}$$

$$\mathbf{x}_i^\top \mathbf{A} \mathbf{x}_j = 0 \quad i \neq j \tag{2.50}$$

❖ 調整向量使得

$$\mathbf{y}_i^\top \mathbf{x}_i = 1 \tag{2.51}$$

$$\mathbf{y}_i^\top \mathbf{A} \mathbf{x}_i = \lambda_i \tag{2.52}$$

2.16 迭代法解特徵值問題

❖ 第一個方法求主要的或最大的特徵值。此法稱為乘方法(power method) 或矩陣迭代。

❖ 考慮(2.37) 式所定義的特徵值問題，並令向量$\mathbf{u}_0$是一初始嘗試解。

❖ 向量$\mathbf{u}_0$ 是所有特徵向量的線性組合如下：

$$\mathbf{u}_0 = \sum_{i=1}^{n} \alpha_i \mathbf{x}_i \tag{2.53}$$

❖ 其中α_i 是未知係數且$\mathbf{x}_i$ 是未知的特徵向量。令迭代法是

$$\mathbf{u}_1 = \mathbf{A}\mathbf{u}_0,\ \mathbf{u}_2 = \mathbf{A}\mathbf{u}_1, \ldots, \mathbf{u}_p = \mathbf{A}\mathbf{u}_{p-1} \tag{2.54}$$

❖ (2.53) 式代入(2.54) 式可得

$$\begin{aligned}
\mathbf{u}_1 &= \sum_{i=1}^{n} \alpha_i \mathbf{A}\mathbf{x}_i = \sum_{i=1}^{n} \alpha_i \lambda_i \mathbf{x}_i \quad \text{因} \quad \mathbf{A}\mathbf{x}_i = \lambda_i \mathbf{x}_i \\
\mathbf{u}_2 &= \sum_{i=1}^{n} \alpha_i \lambda_i \mathbf{A}\mathbf{x}_i = \sum_{i=1}^{n} \alpha_i \lambda_i^2 \mathbf{x}_i \\
&\cdots\cdots\cdots\cdots\cdots\cdots \\
\mathbf{u}_p &= \sum_{i=1}^{n} \alpha_i \lambda_i^{p-1} \mathbf{A}\mathbf{x}_i = \sum_{i=1}^{n} \alpha_i \lambda_i^{p} \mathbf{x}_i
\end{aligned} \tag{2.55}$$

❖ 方程式可排列成

$$\mathbf{u}_p = \lambda_1^p \left[\alpha_1 \mathbf{x}_1 + \sum_{i=2}^{n} \alpha_i \left(\frac{\lambda_i}{\lambda_1} \right)^p \mathbf{x}_i \right] \tag{2.56}$$

❖ 將矩陣的n 個特徵值排序成

$$|\lambda_1| > |\lambda_2| > \cdots > |\lambda_n|$$

$$\left[\frac{\lambda_i}{\lambda_1} \right]^p \xrightarrow[p \to \infty]{} 0$$

▶ 數值方法－使用MATLAB程式語言(第三版)

2-1
2-2
2-3
2-4
2-5
2-6
2-7
2-8
2-9
2-10
2-11
2-12
2-13
2-14
2-15
2-16
2-17

❖ 當p 較大時，由(2.56)式可得

$$\mathbf{u}_p \Rightarrow \lambda_1^p \alpha_1 \mathbf{x}_1$$

❖ 於是$\mathbf{u}_p$ 變成正比於$\mathbf{x}_1$ 且相對的$\mathbf{u}_p$ 及$\mathbf{u}_{p-1}$ 之比值趨近λ_1。

❖ 通常每次迭代後，產生的嘗試向量除以其最大元素而將之規一化，於是將向量內的最大元素化減為一。數學表示為

$$\left.\begin{aligned} \mathbf{v}_p &= \mathbf{A}\mathbf{u}_p \\ \mathbf{u}_{p+1} &= \left(\frac{1}{\max(\mathbf{v}_p)}\right)\mathbf{v}_p \end{aligned}\right\} \quad p = 0, 1, 2, \ldots \qquad (2.57)$$

❖ 其中max($\mathbf{v}_p$) 是$\mathbf{v}_p$ 最大係數元素。

❖ 以下的MATLAB 函數eigit 完成迭代法求主要特徵值及相關的特徵向量。

▶ 數值方法－使用MATLAB程式語言(第三版)

```
function [lam u iter] = eigit(A,tol)
% Solves EVP to determine dominant eigenvalue and associated vector
% Sample call: [lam u iter] = eigit(A,tol)
% A is a square matrix, tol is the accuracy
% lam is the dominant eigenvalue, u is the associated vector
% iter is the number of iterations required
[n n] = size(A);
err = 100*tol;
u0 = ones(n,1);  iter = 0;
while err>tol
    v = A*u0;
    u1 = (1/max(v))*v;
    err = max(abs(u1-u0));
    u0 = u1;  iter = iter+1;
end
u = u0;  lam = max(v);
```

2-1 2-2 2-3 2-4 2-5 2-6 2-7 2-8 2-9 2-10 2-11 2-12 2-13 2-14 2-15 2-16 2-17

2-1
2-2
2-3
2-4
2-5
2-6
2-7
2-8
2-9
2-10
2-11
2-12
2-13
2-14
2-15
2-16
2-17

❖ 求下列特徵值問題的主要特徵值及其特徵向量。

$$\begin{bmatrix} 1 & 2 & 3 \\ 2 & 5 & -6 \\ 3 & -6 & 9 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ x_3 \end{bmatrix} = \lambda \begin{bmatrix} x_1 \\ x_2 \\ x_3 \end{bmatrix} \tag{2.58}$$

2-1
2-2
2-3
2-4
2-5
2-6
2-7
2-8
2-9
2-10
2-11
2-12
2-13
2-14
2-15
2-16
2-17

```
>> A = [1 2 3;2 5 -6;3 -6 9];
>> [lam u iterations] = eigit(A,1e-8)

lam =
   13.4627

u =
    0.1319
   -0.6778
    1.0000

iterations =
    18
```

❖ 迭代法也可以用來求系統的最小特徵值，特徵值問題$\mathbf{Ax}=\lambda\mathbf{x}$ 重新排列成

$$\mathbf{A}^{-1}\mathbf{x} = (1/\lambda)\mathbf{x}$$

❖ 第二種迭代程序，稱為逆迭代，是求次要特徵解(subdominant eigensolutions) 的有效方法。再次考慮(2.37) 式的特徵值問題，將方程式兩邊減去$\mu\,\mathbf{x}$可得

$$(\mathbf{A} - \mu\mathbf{I})\mathbf{x} = (\lambda - \mu)\mathbf{x} \tag{2.59}$$

$$(\mathbf{A} - \mu\mathbf{I})^{-1}\mathbf{x} = \left(\frac{1}{\lambda - \mu}\right)\mathbf{x} \tag{2.60}$$

❖ 考慮這一迭代方法以嘗試向量$\mathbf{u}_0$開始，然後使用等效於(2.57) 式的方式可得

$$\left.\begin{aligned} \mathbf{v}_s &= (\mathbf{A} - \mu\mathbf{I})^{-1}\mathbf{u}_s \\ \mathbf{u}_{s+1} &= \left(\frac{1}{\max(\mathbf{v}_s)}\right)\mathbf{v}_s \end{aligned}\right\} \quad s = 0, 1, 2, \ldots \tag{2.61}$$

❖ 收斂完成時

$$\frac{1}{\lambda-\mu}=\max(\mathbf{v}_s)$$

❖ 所以最靠近λ 之μ 值是

$$\lambda=\mu+\frac{1}{\max(\mathbf{v}_s)} \tag{2.62}$$


```
function [lam u iter] = eiginv(A,mu,tol)
% Determines eigenvalue of A closest to mu with a tolerance tol.
% Sample call: [lam u] = eiginv(A,mu,tol)
% lam is the eigenvalue and u the corresponding eigenvector.
[n,n] = size(A);
err = 100*tol;
B = A-mu*eye(n,n);
u0 = ones(n,1);
iter = 0;
while err>tol
    v = B\u0; f = 1/max(v);
    u1 = f*v;
    err = max(abs(u1-u0));
    u0 = u1; iter = iter+1;
end
u = u0; lam = mu+f;
```

```
>> A = [1 2 3;2 5 -6;3 -6 9];
>> [lam u iterations] = eiginv(A,4,1e-8)

lam =
    4.1283

u =
    1.0000
    0.8737
    0.4603

iterations =
     6
```


▶數值方法－使用MATLAB程式語言(第三版)

2.17 MATLAB 函數eig

❖ 現在說明MATLAB 函數eig 使用的演算法。同時在這程序使用不同的演算法如下：

1. lambda=eig(a)

2. [u,lambda]=eig(a)

3. lambda=eig(a,b)

4. [u,lambda]=eig(a,b)

❖ QR 程序包括分解海森伯格矩陣成上三角矩陣與單式矩陣(unitary matrix)，方法如下：

1. $k=0$

2. 分解$\mathbf{H}_k$成$\mathbf{Q}_k$和$\mathbf{R}_k$使得$\mathbf{H}_k=\mathbf{Q}_k\mathbf{R}_k$，這裡$\mathbf{H}_k$是一海森伯格或三對角線(tridiagonal) 矩陣。

3. 計算$\mathbf{H}_k+1=\mathbf{Q}_k\mathbf{R}_k$，特徵值的估側等於 diag($\mathbf{H}_k$+1)

4. 檢查特徵值的精確度。若這程序尚未收斂，$k=k+1$ 然後重覆(2)。

▶ 數值方法－使用MATLAB程式語言(第三版)

2-1
2-2
2-3
2-4
2-5
2-6
2-7
2-8
2-9
2-10
2-11
2-12
2-13
2-14
2-15
2-16
2-17

❖ 例如：

```
% e3s213.m
A = [5 4 1 1;4 5 1 1; 1 1 4 2;1 1 2 4];
H1 = hess(A);
for i = 1:10
     [Q R] = qr(H1);
    H2 = R*Q;  H1 = H2;
    p = diag(H1)';
    fprintf('%2.0f %8.4f %8.4f',i,p(1),p(2))
    fprintf('%8.4f %8.4f\n',p(3),p(4))
end
```

1	1.0000	8.3636	6.2420	2.3944
2	1.0000	9.4940	5.4433	2.0627
3	1.0000	9.8646	5.1255	2.0099
4	1.0000	9.9655	5.0329	2.0016
5	1.0000	9.9913	5.0084	2.0003
6	1.0000	9.9978	5.0021	2.0000
7	1.0000	9.9995	5.0005	2.0000
8	1.0000	9.9999	5.0001	2.0000
9	1.0000	10.0000	5.0000	2.0000
10	1.0000	10.0000	5.0000	2.0000

- 當有2 實數或複數自變數(arguments) 在MATLAB函數eig 內時，使用QZ 演算法而非QR 演算法。
- QZ 演算法由注意存在一單式矩陣**Q** 和**Z** 使**Q**^H^**AZ**=**T**和**Q**^H^**BZ**=**S** 皆為上三角矩陣開始。這稱為廣義舒爾分解 (generalized Schurd ecomposition)。假如s_{kk} 不為0，則特徵值由 t_{kk}/s_{kk} 計算出，其中$k=1,2,\cdots n$。

```
% e3s214.m
A = [10+2i 1 2;1-3i 2 -1;1 1 2];
b = [1 2-2i -2;4 5 6;7+3i 9 9];
[T S Q Z V] = qz(A,b);
r1 = diag(T)./diag(S)
r2 = eig(A,b)
```

```
r1 =
   1.6154 + 2.7252i
  -0.4882 - 1.3680i
   0.1518 + 0.0193i

r2 =
   1.6154 + 2.7252i
  -0.4882 - 1.3680i
   0.1518 + 0.0193i
```


❖ **A 可寫成**

$$A = UTU^H$$

```
% e3s215.m
A = [4 -5 0 3;0 4 -3 -5;5 -3 4 0;3 0 5 4];
T = schur(A), lam = eig(A)
```

```
T =
   12.0000    0.0000   -0.0000   -0.0000
         0    1.0000   -5.0000   -0.0000
         0    5.0000    1.0000   -0.0000
         0         0         0    2.0000

lam =
  12.0000
   1.0000 + 5.0000i
   1.0000 - 5.0000i
   2.0000
```

❖ 以下程式比較eig 函數解不同問題種類的性能。

```
% e3s216.m
disp('        real1    realsym1      real2     realsym2      comp1       comp2')
for n = 100:50:500
    A = rand(n); C = rand(n);
    S = A+C*i;
    T = rand(n)+i*rand(n);
    tic, [U,V] = eig(A); f1 = toc;
    B = A+A.'; D = C+C.';
    tic, [U,V] = eig(B); f2 = toc;
    tic, [U,V] = eig(A,C); f3 = toc;
    tic, [U,V] = eig(B,D); f4 = toc;
    tic, [U,V] = eig(S); f5 = toc;
    tic, [U,V] = eig(S,T); f6 = toc;
    fprintf('%12.3f %10.3f %10.3f %10.3f %10.3f %10.3f\n',f1,f2,f3,f4,f5,f6);
end
```


real1	realsym1	real2	realsym2	comp1	comp2
0.042	0.009	0.063	0.061	0.039	0.037
0.067	0.014	0.086	0.090	0.067	0.106
0.129	0.028	0.228	0.184	0.116	0.200
0.182	0.046	0.430	0.425	0.186	0.432
0.270	0.073	0.729	0.724	0.279	0.782
0.371	0.104	1.277	1.257	0.373	1.232
0.514	0.154	2.006	2.103	0.538	2.104
0.708	0.205	3.055	3.097	0.698	2.919
0.946	0.278	4.403	4.187	0.901	4.344

❖ MATLAB 也包括求稀疏矩陣特徵值的設備。

```
% e3s217.m
% generate a sparse triple diagonal matrix
n = 2000;
rowpos = 2:n; colpos = 1:n-1;
values = ones(1,n-1);
offdiag = sparse(rowpos,colpos,values,n,n);
A = sparse(1:n,1:n,4*ones(1,n),n,n);
A = A+offdiag+offdiag.';
% generate full matrix
B = full(A);
tic, eig(A); sptim = toc;
tic, eig(B); futim = toc;
fprintf('Time for sparse eigen solve = %8.6f\n',sptim)
fprintf('Time for  full  eigen solve = %8.6f\n',futim)
```

```
Time for sparse eigen solve = 0.349619
Time for  full  eigen solve = 3.000229
```